2003-05-05

NPSAT1 attitude control subsystem hardware-in-the-loop simulation

Schmidt, Alexander

Monterey, California. Naval Postgraduate School

NPS-SP-03-002

NAVAL POSTGRADUATE SCHOOL
Monterey, California

NPSAT1 Attitude Control Subsystem Hardware-in-the-Loop Simulation

by

Alexander Schmidt

5 May 2003

Prepared for: NPS Space Systems Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

NAVAL POSTGRADUATE SCHOOL

Monterey, California 93943-5000

RADM David Ellison
Superintendent

R. Elster
Provost

This report was prepared for and funded by the Naval Postgraduate School, Space Systems Academic Group.

This report was prepared by:

Alexander Schmidt
Oberleutnant, German Army

Reviewed by:

Rudolf Panholzer
Space Systems Academic Group

Released by:

D. W. Netzer
Associate Provost and
Dean of Research

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** May 2003	**3. REPORT TYPE AND DATES COVERED** Technical Report	
4. TITLE AND SUBTITLE: Title (Mix case letters) NPSAT1 Attitude Control Subsystem Hardware-in-the-Loop Simulation			**5. FUNDING NUMBERS** N/A
6. AUTHOR(S) Alexander Schmidt, Oberleutnant German Army			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Space Systems Academic Group Code (SP), 777 Dyer Rd., Rm 200 Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER** NPS-SP-03-002
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Naval Postgraduate School Space Systems Academic Group Code (SP), 777 Dyer Rd., Rm 200 Monterey, CA 93943-5000			**10. SPONSORING / MONITORING AGENCY REPORT NUMBER** N/A
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release, distribution is unlimited.			**12b. DISTRIBUTION CODE**

13. ABSTRACT ***(maximum 200 words)***

NPSAT1 is a three-axis stabilized spacecraft. Its Attitude Control Subsystem (ACS) uses a magnetic control approach that will be used for the first time. The Magnetic control approach is verified with an ACS SIMULINK model of NPSAT1. The correct SIMULINK implementation of the magnetic control algorithm will be verified with an ACS air bearing SIMULINK model and a hardware-embedded ACS control algorithm SIMULINK model that controls the test platform on a spherical air bearing table.

This is a report of the work that covers different steps of the air bearing table setup for these hardware-in-the-loop simulations. The first part describes the approach of determining an air bearing table location in Earth's magnetic field with minimal magnetic deviation to provide good conditions for hardware-in-the-loop simulations. The second part is the attempt of developing custom driver software for the magnetic measurement device that is going to be used on the air bearing test platform. The third part gives some information about the general structure of the hardware-in-the-loop test setup and the hardware setup of the air bearing platform.

14. SUBJECT TERMS Attitude Control Subsystem (ACS), Magnetic Torque Rods, NPSAT1			**15. NUMBER OF PAGES** 157
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The *Naval Postgraduate School Spacecraft Architecture and Technology Demonstration Experiment* (NPSAT1) is a low cost, technology demonstration satellite hosting a number of experiments.

NPSAT1 is a three-axis stabilized spacecraft. Its Attitude Control Subsystem (ACS) uses a magnetic control approach that will be used for the first time. Attitude errors are determined by comparing actual magnetic environmental data (measured in a body coordinate system) with known magnetic environmental data (given in a orbit reference coordinate system). The magnetic control algorithm determines the components of a magnetic dipole moment that has to be applied to interact with the Earth's magnetic field. The resulting magnetic torque is used to control the attitude of NPSAT1.

The magnetic control approach is verified with an ACS SIMULINK® Model of NPSAT1. The correct SIMULINK® implementation of the magnetic control algorithm will be verified with an ACS air-bearing SIMULINK® model and hardware-embedded ACS control algorithm SIMULINK® that controls the test platform on a spherical air-bearing table. Both models use the same implementation of the magnetic control algorithm. These tests are hardware-in-the-loop simulations.

This is a report of the work that covers different steps of the air-bearing table setup for these hardware-in-the-loop simulations. It is based on a previous Interdisziplinäre Studienarbeit *NPSAT1 Hardware-in-the-Loop Simulation. Design of the Air-bearing Platform*, [Ref. 3].

First a reasonable location for the air-bearing table was determined. The actual magnetic flux density field in the Space Systems Academic Group laboratory was measured. After that the amount of data was reduced and the magnetic flux density vector field was visualized / mapped. The challenge was the determination of a

location with small field deviations to ensure good magnetic environmental conditions for the hardware-in-the-loop simulations. Different approaches were considered to find a useful reasonable field deviation measure. After some setbacks a way was found to compare the information given by a set of five magnetic flux density field vectors surrounding one measurement point with information given by each other set of five field vectors of the measured actual magnetic field. The resulting air-bearing position was confirmed with information obtained by other approaches. The obtained measurement information can be used for future magnetic field measurements since the air-bearing table has to be moved because of reconstruction work at the SSAG Laboratory.

The next step was the task to develop a custom made magnetometer driver. The magnetometer that was used for measurements of the magnetic field is intended to be used as the measurement device in hardware-in-the-loop simulation on the air-bearing platform. Before a MATLAB SIMULINK® implementation was considered, a LabVIEW^{TM} driver was developed to obtain information and experience in controlling the magnetometer. Developing a MATLAB SIMULINK® magnetometer driver was not that easy and is not accomplished yet. This has several reasons. The embedded ACS control algorithm SIMULINK® model is compiled into an xPC Target real-time application and runs on a single computer board. This approach necessitates the implementation of communication features within the SIMULINK® model to create communication interfaces when this model is compiled into the xPC Target real-time application. One major problem is still the incompatibility of MATLAB SIMULINK® supported output response format and the magnetometer output response format. Different attempts were done to solve this problem but nothing has worked yet. The obtained information may be useful for a future solution.

Work on the hardware air-bearing table setup was blocked unfortunately by different problems concerning the construction of the air-bearing platform and its electronic components. Thats why work was concentrated on solving the task of

a MATLAB SIMULINK® magnetometer driver. Even intensified work with Jim Horning, SSAG software engineer, has not led to encouraging results yet. However some considerations about the air-bearing platform setup for the hardware-in-the-loop simulation were done.

One goal of SSAG is to provide an opportunity to participate in developing and building an actual space system. This very complex process is characterized by team work, interdependence and experience of different approaches of problem solving. Since the ACS SIMULINK® model hardware-in-the-loop simulation is a new approach little information exists. This offered the opportunity of experiencing the reality of systems engineering.

THIS PAGE INTENTIONALLY LEFT BLANK

ZUSAMMENFASSUNG

Das *Naval Postgraduate School Spacecraft Architecture and Technology Demonstration Experiment* (NPSAT1) ist ein niedrigkosten, technologiedemonstrierender Satellit, der eine Reihe von Experiementen beherbergt.

NPSAT1 wird ein drei-achsen stabilisierter Satellit sein, dessen Fluglagenregelungssystem, Attitude Control Subsystem (ACS), auf einem magnetischen Regelungsprinzip beruht und das erste Mal überhaupt praktisch umgesetzt wird. Die Fluglagenabweichungen werden wie folgt bestimmt. NPSAT1 misst Daten des umgebenden Magnetfeldes in seinem Körperkoordinatensystem und vergleicht sie mit bekannten Magnetfelddaten, die im Orbit-Referenz Koordinatensystem gegeben sind. Der Regelalgorithmus bestimmt aufgrund dieser Fluglagenabweichungen die Komponenten eines magnetischen Dipolvektors. Wird dieser magnetische Dipolvektor vom Satellit erzeugt, dann entsteht in Zusammenwirken mit dem Erdmagnetfeld ein Drehmoment, das zur Fluglagenregelung eingesetzt werden kann.

Die Umsetzbarkeit dieses magnetischen Fluglagenregelungsprinzips wurde mit einem SIMULINK® Modell von NPSAT1 nachgewiesen. Um nun nachzuweisen, dass der Regelagorithmus dieses magnetischen Fluglagenregelungsprinzips richtig im SIMULINK® Modell implementiert ist, wurde ein abgeändertes SIMULINK® Modell einer luftgelagerten Veruchsstrucktur erstellt. Der Regelalgorithmus (das ist derselbe wie im originalen SIMULINK® Modell) dieses Luftlager- SIMULINK® Modells wird in voll funktionsfähige Hardware eingebettet und regelt die luftgelagerte Versuchsstrucktur. Der Vergleich von Luftlager-SIMULINK® Modell Simulationsdaten mit gemessenen Daten der real geregelten Versuchsstrucktur wird die richtige Implementierung des Regelalgorithmus nachweisen. Das sind die sogenannten hardware-in-the-loop Simulationen.

Der vorliegende Bericht beschreibt die durchgeführten Arbeiten zu diesem Thema, die mehrere Abschnitte des Prozesses umfassen, die lufgtgelagerte Versuchsstrucktur für die hardware-in-the-loop Simulationen vorzubereiten. Diese Arbeiten bauen auf der Interdisziplinären Studienarbeit *NPSAT1 Hardware-in-the-Loop Simulation. Design of the Air-bearing Platform*, [Ref. 3], auf.

Die erste Notwendigkeit war, eine geeignete Position für das Luftlager zu ermitteln. Dazu wurde das tatsächlich im SSAG Labor vorhandene Magnetfeld vermessen. Um die Ergebnisse geeignet darstellen zu können, wurden die Daten angemessen reduziert. Die nächste Herausforderung war, eine Position mit möglichst geringer Magnetfeldänderung zu finden, um sicherzustellen, das ein für Testzwecke geeignetes magnetisches Umfeld vorliegt. Deswegen wurden verschiedene Methoden ausprobiert, um ein nützliches Entscheidungskriterium zu finden. Trotz einiger Rückschläge wurde eine Möglichkeit gefunden, die Informationen einer Gruppe von fünf Magnetfeldvektoren mit denen jeder anderen Fünfergruppe des vermessenen Magnetfeldes zu vergleichen. Eine Fünfergruppe Magnetfeldvektoren beschreibt einen Messpunkt. Die schließlich gewählte Luftlagerposition wurde mit Informationen bestätigt, die mit anderen Methoden erlangt wurden. Die Informationen aus diesem Mess- und Analyseabschnitt können wieder verwendet werden, da der Luflagerstandort bauarbeitenbedingt wechseln muss.

Der nächste Abschnitt war, einen nutzerangepassten Magnetometer Treiber zu entwickeln. Dieses Magnetometer war schon für die Magnetfeldmessungen im vorangegangen Abschnitt verwendet worden und soll jetzt als Messinstrument für die hardware-in-the-loop Simulationen verwendet werden. Um die Arbeit, einen MATLAB SIMULINK® Treiber zu entwickeln, zu erleichtern, sollte ein LabVIEW^{TM} Treiber entwickelt werden. Die so gewonnenen Informationen haben ihren Zweck erfüllt. Dagegen ist die Arbeit am MATLAB SIMULINK® Treiber noch nicht abgeschlossen. Diese Aufgabe hat sich als komplizierter herausgestellt, als es erwartet wurde. Das hat mehrere Gründe. Das eingebettete ACS (Regelalgorith-

mus) SIMULINK® Modell wird in eine xPC Target Echtzeit Anwendung übersetzt und läuft auf einem eigenen Computerboard. Dieses Vorgehen verlangt, dass die Software-Werkzeuge, die später die Schnittstellen für die Kommunikation werden, bereits im SIMULINK® Modell implementiert werden. Das bedeutet, dass SIMULINK® unterstützte Werkzeuge genutzt werden müssen. Eine der noch ungelösten Schwierigkeiten ist, dass das Format der Magnetometer Daten nicht zu den von SIMULINK® unterstützen, bzw. verlangten, Formaten passt. Verschiedene Möglichkeiten wurden versucht aber noch keine erfolgreiche gefunden. Hoffnung besteht, dass die nach dem Ausschlussprinzip erhaltenen Informationen zu einer Problemlösung beitragen.

Gleichzeitig wurde versucht, den Versuchsaufbau des Luftlagers voranzutreiben. Unglücklicherweise wurde das von verschiedenen Problemen mit der Herstellung der luftgelagerten Versuchsstrucktur und der nötigen Regelelektronik behindert und ist ebenfalls noch nicht abgeschlossen. Aufgrund der zwischenzeitlichen Behinderungen wurde die Zusammenarbeit mit Jim Horning, Software-Ingenieur der SSAG, intensiviert, um den SIMULINK® unterstützten Treiber fertigzustellen. Trotzdem liegen noch keine unmittelbar für die hardware-in-the-loop Simulationen nutzbaren Ergebnisse vor. Die Arbeit auf diesem Gebiet wird noch fortgesetzt.

Eines der Ziele der Space Systems Academic Group ist, Arbeit auf dem Gebiet der Entwicklung von realen Raumfahrzeugen anzubieten. Dieser sehr komplexe Prozess ist geprägt von Team Arbeit, gegenseitiger Abhängingkeit und unterschiedlichsten Varianten der Problembewältigung. Auch für die SSAG ist der Aufbauprozess der hardware-in-the-loop Simulationen neu. Das hat letztendlich die wertvollen Erfahrungen der Realität in dem weiten Bereich der Ingenieurtätigkeit ermöglicht.

THIS PAGE INTENTIONALLY LEFT BLANK

DECLARATION

Hereby I declare that this Thesis has been created by myself using only the named sources and supplementary equipment.

ERKLÄRUNG

Hiermit erkläre ich, dass die vorliegende Arbeit von mir selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel erstellt wurde.

Monterey, 23 April 2003

Alexander Schmidt, OL

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS, ACRONYMS AND/OR ABBREVIATIONS

space-engineering based abbreviations

ACS	Attitude Control Subsystem
CERTO	Coherent Electromagnetic Radio Tomography
C&DHS	Command and Data Handling Subsystem
COTS	Commercial, off the Shelf
CPE	Configurable Processor Experiment
EELV	Evolved Expandable Launch Vehicle
EPS	Electrical Power Subsystem
ESPA	Evolved Expandable Launch Vehicle Secondary Payload Adapter
GG	Gravity Gradient
NPSAT1	Naval Postgraduate School Spacecraft Architecture and Technology Demonstration Experiment 1
PANSAT	Petite Amateur Navy Satellite
RFS	Radio Frequency Subsystem
SSAG	Space Systems Academic Group
VISIM	Visible Wavelength Imager

other

BCD ASCII	Binary Coded Decimal American Standard Code for Information Interchange
FS	Full Scale
NI	National Instruments

PCA	Principle Component Analysis
RS232	Recommended Standard number 232
SI	Système International
VISA	Virtual Instrument Software Architecture

symbols

$\mathbf{A}_{\phi\theta\psi}$	Euler Angle rotation transformation matrix, $[\mathrm{A}_{\phi\theta\psi,ij}] = /$
$\mathbf{B}$	vector of magnetic flux density, $[\mathrm{B}_i] = \text{Tesla } (T)$
d_0	supposed difference of two Expected Values
$E(.)$	expected value of (.)
$f(x)$	distribution density function, derived from empirical distribution func.
$F(x)$	empirical distribution function
g	acceleration of fall, $[g] = ms^{-2}$
$\mathbf{I}$	tensor of (moments and products of) inertia, $[\mathrm{I}_{ij}] = kgm^2$
$i,\ j,\ k$	indices
m	mass, $[m] = kg$
$\mathbf{m}$	vector of magnetic dipole moment, $[\mathrm{m}_i] = Am^2$
N	number of values, e.g. in one sample $\mathbf{X}$
$\mathbf{r_{cm}}$	distance vector of center of rotation and center of mass, $[\mathrm{r}_{\mathrm{cm,i}}] = m$
$\mathbf{r}_{i,k}$	vector, pointing from a location i in direction of k
s	standard deviation
s^2	empirical variance
t	quantile of Student's distribution, $[t] = /$
$\mathbf{T}$	torque vector, $[\mathrm{T}_i] = Nm$
$\mathbf{T_c}$	vector of control torques, $[\mathrm{T}_{c,i}] = Nm$
$\mathbf{T_d}$	vector of disturbance torques, $[\mathrm{T}_{d,i}] = Nm$
u	quantile of Gaussian distribution function, $[u] = /$
x_i	measured value, component of $\mathbf{X}$, element of one population

$\mathbf{X}$	sample, vector of measured values
$\bar{x}$	empirical mean
$\tilde{x}$	empirical median
$x,\ y,\ z$	Cartesian coordinates of a *reference coordinate system*
α	probability of error
δ	difference of two Expected Values
μ	Expected Value, 1st moment of a distribution function
σ^2	Variance, 2nd moment of a distribution function
$\varphi(x)$	distribution density function
$\Phi(x)$	distribution function
$\phi,\ \theta,\ \psi$	Euler Angles, $[\ ] = rad$
$\xi,\ \eta,\ \zeta$	Cartesian coordinates of a *body coordinate system*
ω_0	angular orbital velocity, $[\omega_0] = rad/s$
$\boldsymbol{\omega}$	vector of angular velocity, $[\omega_i] = rad/s$
$C.L.$	confidence level
$C.I.$	confidence intervall
$s(.)$	sin(.)
$c(.)$	cos(.)
$\dot{(.)}$	derivative of (.) with respect to time
$\ddot{(.)}$	second derivative of (.) with respect to time
$\lvert . \rvert$	norm of (.)
$\sphericalangle(i,k)$	angle between i and k

I have tried to use a consistent statistical nomenclature. Greek letters are used to name values that are characteristics of a whole population - $\mu,\ \sigma^2,\ \Phi(x),\ \varphi(x)$. Latin letters are used to name values that are characteristics of one sample (statistical/empirical values) - $\bar{x},\ s^2,\ F(x), f(x)$.

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGEMENT

I would like to thank Professor Rudolf Panholzer and the Space Systems Academic Group which have supported my work with this challenging topic of NPSAT1 hardware-in-the-loop simulation and have let me experience a part of the process of developing a spacecraft. I want to thank Professor Hendrik Rothe and Professor Reinhart Lunderstädt which have offered me the opportunity to write this Diplomarbeit at the US Naval Postgraduate School, Monterey, California.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

This chapter introduces the topic of *Hardware-in-the-loop Simulation* of the ACS SIMULINK® Model. First a short overview of the structure of this report is given. The NPSAT1 project is briefly described and later on the magnetic attitude control approach, that is one of the main experiments. A first step to success of that experiment is the verification of a correct implementation of the *Attitude Control Subsystem* (ACS) SIMULINK® Model. This is the task of the hardware-in-the-loop simulations.

A. STRUCTURE

Chapter I. As mentioned above this first Chapter is used to explain the structure of this report. Furthermore are given also some information about the NPSAT1 project.

Chapter II. Chapter II describes the measurement of the actual, current magnetic flux density field (**B** field) in the SSAG laboratory that was intended to be the testing environment of the embedded ACS (control algorithm) SIMULINK® Model on a spherical air-bearing table. Since the SSAG building is under reconstruction now, the position of the air-bearing table will change. Different advanced considerations have shown what data analysis approaches lead to reasonable results. One may use information gained with these measurements as a guideline for measurements and analysis. The measured data are contained on the data CD of this report.

Chapter III. In Chapter III is described the attempt of developing different custom made magnetometer drivers. The decision to use a different magnetometer than the one intended has necessitated a different approach of communication with this measurement device. It is intended to replace the Schonstedt Instrument CO. SAM-73C magnetometer that necessitates analog/didital conversion with a Honeywell HMR2300 smart digital magnetometer that uses serial communication.

Chapter IV The topics of Chapter IV are considerations about the air-bearing platform setup. The air-bearing platform design process is described in [Ref. 8]. This air-bearing platform will be used in hardware-in-the-loop simulations of a SIMULINK® implemented magnetic control approach that is not in use yet. Furthermore some information about the assembling of the air-bearing platform is presented. The goal of finishing the whole hardware-in-the-loop simulation setup is not reached yet.

Chapter V. The last chapter reflects shortly the work that was done.

Appendix A. Appendix A contains tables with technical specifications of the magnetometer HMR2300. General specifications as well as programming information are shown.

Appendix B. In Appendix B are placed Maple 7.0 sequences that were used to analyze the Earth's magnetic field data in Chapter II. Some of these sequences may be used for analysis of measured data to determine the next (final) position of the air-bearing table. The actual Maple 7.0 scripts are contained on the data CD. Furthermore some visualizations of the measured data are shown.

Appendix C. Appendix C shows the experimental SIMULINK® models and their settings that were used trying to establish accurate serial communication with the HMR2300 magnetometer.

Appendix D. An overview of all technical drawings of air-bearing platform components is given in Appendix D. The actual technical drawings are also contained on the data CD.

Appendix E. The Maple 7.0 work sheet for estimating the mass properties of the air-bearing table is explained in Appendix E. This Maple 7.0 work sheet is also part of the data CD. The Maple 7.0 work sheets that determine the moments of inertia of the hexagonal base plate and of the angle stock material can also be found on that CD.

B. NPSAT1 PROJECT OVERVIEW

1. General

The *Naval Postgraduate School Spacecraft Architecture and Technology Demonstration Experiment* (NPSAT1) is the second spacecraft built by the *Space Systems Academic Group* (SSAG) at the US Naval Postgraduate School (*NPS*) in Monterey, California. It is the follow-on project of *Petite Amateur Navy Satellite* (PANSAT). NPSAT1 is a small satellite and manifested on the Department of Defense Space Test Programm STP-1, Delta IV mission, due to launch in March 2006.

The SSAG is an interdisciplinary group. It offers a wide spectrum of thesis topics to student officers with different scientific background [Ref. 1]. NPSAT1 is a spacecraft built by officer students, faculty and staff of the NPS. It offers to experience of the whole process of developing and building a spacecraft and the reality of systems engineering.

NPSAT1 is a low-cost technology demonstration satellite. It hosts a number of different experiments. One of these experiments is the *Attitude Control Subsystem* (ACS). This ACS uses a magnetic attitude control approach that is a topic for theoretical studies but not applied in practice yet. NPSAT1 will include commercial-off-the-shelf (COTS) technology in many of its subsystems.

2. NPSAT1 Experiments & Estimated Structural Properties

NPSAT1 is a cylindrical spacecraft. Its major dimensions and estimated mass properties are shown in Table I. NPSAT1 components will be located on three different shelves within the spacecraft. Many of these components are part of a number of different experiments.

C&DHS / RFS. The *Command and Data Handling Subsystem* and the *Radio Frequency Subsystem* are used to collect data to download to the ground station and

Dimension	*Value*
major diameter: hull deployed booms	 $498.4mm$ $1490.3mm$
height (over all)	$841.3mm$
estimated Mass Property	*Value*
mass	$81.7kg$
$I\eta\eta$ $I_{\xi\xi}$ $I_{\zeta\zeta}$	$5.2kgm^2$ $5.0kgm^2$ $2kgm^2$

Table I. Estimated General NPSAT1 Specifications.

communicate with the ground station. The C&DHS coordinates and manages the functions of the spacecraft and turns on and off experiments. It will consist of COTS-based technology.

EPS. The *Electrical Power Subsystem* provides the components of the spacecraft with electrical power. It converts energy by means of experimental, improved, triple junction technology solar cells (part of the a solar cell measurement system experiment) and commercial grade, improved, triple junction cells. It stores energy in Lithium-Ion batteries.

ACS. The *Attitude Control Subsystem* is another experiment. It uses a magnetic control approach that will be implemented for the first time although the theory has existed already. As it is subject to the hardware-in-the-loop simulation it will be described in more detail in Paragraph I.B.3.

Nonvolatile ferroelectric RAM. This RAM is inherently radiation tolerant. NPSAT1 demonstrates the use.

CERTO / Langmuir Probe. The *Coherent Electromagnetic Radio Tomography* experiment is for measuring the integrated electron density of the ionosphere in the observation plane together with a network of ground receivers. The Langmuir probe serves the same purpose.

CPE / VISIM. The *Configurable Processor Experiment* acts firstly as triple-modular redundant computer that corrects detected single event upsets within the procession without rebooting the processor. Secondly is used to implement a hardware compression engine to produce jpeg representations of VISIM data. The *Visible Wavelength Imager* is a COTS digital camera.

3. Attitude Control Subsystem

NPSAT1 will be a nadir-pointing satellite. This means that the outer normal vector of its bottom is intended to point always to Earth's center. As the configuration of a satellite can be described in three *Euler Angles* - ϕ, θ, ψ -, this nadir-pointing demands the Euler Angles are kept as small as possible (see [Ref. 3]). An Attitude Control Subsystem facilitates this.

Different disturbances act on a spacecraft in orbit. The current attitude of NPSAT1 is the result of aero disturbance torques in low earth orbit and control torques etc. Disturbance torques may cause position errors of the spacecraft or excite oscillations. Control torques are applied to control the attitude of such a spacecraft.

There are different attitude control approaches such as pure passive *Gravity Gradient* (GG) stabilization or GG stabilization with passive and active oscillation damping (e.g. magnetic damping) as well as active stabilization. Magnetic damping is already in use. But the use of magnetic control torques interacting with the Earth's magnetic field to control the attitude of a spacecraft is not used in practice yet, [Ref. 3].

The principle of the used magnetic control approach seems to be not that difficult. But there are some difficulties. The components of a requested magnetic dipole moment have to be determined from a vectorial cross product. The components of the resulting applied magnetic control torque influence more than one Euler Angle, etc.

The idea of the used magnetic control approach is briefly described. NPSAT1 will have a GG friendly design. That means

$$I_{\eta\eta} > I_{\xi\xi} > I_{\zeta\zeta}, \tag{I.1}$$

and GG stabilization supports attitude control [Ref. 3]. An on-board orbit propagator determines the position of NPSAT1. The Earth's magnetic field vector is calculated by means of the eight-order *International Geomagnetic Reference Field* (IGRF, see [Ref. 4]) for a series of orbit positions and stored in a look-up table. If the position of the spacecraft is known, the theoretical magnetic field vector can be determined in *orbit reference coordinates* - $\mathbf{B}_{[\mathbf{xyz}]}$. A magnetometer measures simultaneously the current magnetic field vector $\mathbf{B}_{[\xi\eta\zeta]}$ in *body coordinates*. The attitude errors can be expressed in Euler Angles and determined by the Euler Angle rotation $\mathbf{A}_{\phi\theta\psi}$ (see [Ref. 3],[Ref. 8]):

$$\mathbf{B}_{[\xi\eta\zeta]} = \mathbf{A}_{\phi\theta\psi}\mathbf{B}_{[\mathbf{xyz}]}. \tag{I.2}$$

The requested attitude control torque $\mathbf{T_c}$ can be determined from the dynamic behavior of NPSAT1, that is the set of differential equations of motion [Ref. 3],[Ref. 7]:

$$\begin{bmatrix} T_{d\xi} + T_{c\xi} \\ T_{d\eta} + T_{c\eta} \\ T_{d\zeta} + T_{c\zeta} \end{bmatrix} = \begin{bmatrix} I_{\xi\xi}\ddot{\phi} + 4\omega_0^2(I_{\eta\eta} - I_{\zeta\zeta})\phi + \omega_0(I_{\eta\eta} - I_{\zeta\zeta} - I_{\xi\xi})\dot{\psi} \\ I_{\eta\eta}\ddot{\theta} + 3\omega_0^2(I_{\xi\xi} - I_{\zeta\zeta})\theta \\ I_{\zeta\zeta}\ddot{\psi} + \omega_0^2(I_{\eta\eta} - I_{\xi\xi})\psi + \omega_0(I_{\zeta\zeta} + I_{\xi\xi} - I_{\eta\eta})\dot{\psi} \end{bmatrix}. \quad \text{(I.3)}$$

This control torque will result from the interaction of a magnetic dipole moment with the Earth's magnetic flux density field. The components of the magnetic dipole moment $\mathbf{m}$ are applied by means of magnetic torque rods. These magnetic dipole moment components can be determined from the following equations:

$$\mathbf{T_c} = \mathbf{m} \times \mathbf{B}_{[\xi\eta\zeta]}, \quad \text{(I.4)}$$

$$\mathbf{m} = \frac{1}{|\mathbf{B}|^2}(\mathbf{B} \times \mathbf{T_c}), \quad \text{(I.5)}$$

with the prerequisite of a perpendicular torque $\mathbf{T_c}$ with respect to the Earth's magnetic field $\mathbf{B}$, [Ref. 3], [Ref. 5].

The ACS can act in two different modes. The first mode is the so called $\dot{\mathbf{B}}$ or **Bdot** control. This mode reduces tip-off rates that result from the launch of the ESPA etc. The second mode is called *pointing mode* and stabilizes and controls the attitude of the spacecraft.

C. ACS SIMULINK® MODEL HARDWARE-IN-THE-LOOP SIMULATION

The hardware-in-the-loop simulation is to test hardware and software of NPSAT1. The ACS of NPSAT1 will consist of the magnetic control approach software and the necessary hardware. Tests of the hardware-in-the-loop simulation will show whether the control algorithm is implemented correctly into the ACS SIMULINK® Model and whether the whole system works properly under real conditions.

1. ACS SIMULINK® Model

M.Kohrt describes in [Ref. 7] the process of embedding a simplified pitch-axis ACS SIMULINK® Model in hardware. The genuine ACS (NPSAT1) SIMULINK® Model is used to verify the feasibility of the magnetic control approach in space. It is a pure computer model of the real world conditions and gets necessary environmental information from data files.

The ACS (NPSAT1) SIMULINK® Model simulation results encourage the use of the magnetic control approach with small satellites like NPSAT1. However, the ACS (NPSAT1) SIMULINK® Model simulation results are just computer simulations and depend on a correct implementation of components of the attitude control subsystem, the magnetic control algorithm and data flow in an artificial computer environment. Outer influences and disturbances can be taken into account only if they are known and mathematical models exist to describe them.

The first step of hardware-in-the-loop simulations was to embed a simplified pitch-axis ACS (control algorithm) SIMULINK® Model into real hardware, to add real input from sensors and real output to actuators and verify this embedded system. The ACS (control algorithm) SIMULINK® Model is the implementation of the magnetic control approach but uses the dynamics of an air-bearing platform instead of the actual spacecraft dynamics. The embedding was done by M.Mohrt, [Ref. 7]. The principle of this embedded system is to have a Matlab SIMULINK® xPC environment running on a host PC that starts, monitors and stops the implemented control algorithm on a target PC. This target PC executes the control algorithm SIMULINK® implementation in real time and is connected to real sensors and actuators.

When M.Kohrt embedded this simplified pitch-axis ACS (control algorithm) SIMULINK® Model it was intended for use within hardware-in-the-loop simulations. The air-bearing table offers the opportunity of free movement not only about one axis but about three axes. Conversely it is very difficult to avoid rotation about the other

two axes with that air-bearing table since the equations of motion are not independent from each other, see Equation (I.3) and additionally [Ref. 3], [Ref. 6]. That is caused by the Euler Angle rotation. The decision was made to use a three axes ACS (control algorithm) SIMULINK® Model instead of the simplified pitch-axis ACS (control algorithm) SIMULINK® Model in [Ref. 7].

That decision necessitates embedding this three axes ACS (control algorithm) SIMULINK® Model on the target PC. This is expected to be done similar to the approach that M.Kohrt has used and described in [Ref. 7]. One major difference is the use of a different magnetometer that facilitates serial communication. Another difference is the desire of serial communication between the target PC application of the control algorithm and the torque rod driver components.

2. Air-bearing Table

The hardware-in-the-loop simulation is used to verify the NPSAT1 ACS SIMULINK® Model. The idea is to replace the equations of motion that are implemented in the NPSAT1 ACS SIMULINK® with equations of motion of an air-bearing Model, (see [Ref. 8]):

$$\begin{bmatrix} T_\xi - r_{cm,\zeta} m s\phi c\theta g \\ T_\eta - r_{cm,\zeta} m s\theta g \\ T_\zeta \end{bmatrix} = \begin{bmatrix} I_{\xi\xi}\dot{\omega}_\xi + (I_{\zeta\zeta} - I_{\eta\eta})\omega_\eta\omega_\zeta \\ I_{\eta\eta}\dot{\omega}_\eta + (I_{\xi\xi} - I_{\zeta\zeta})\omega_\zeta\omega_\xi \\ I_{\zeta\zeta}\dot{\omega}_\zeta + (I_{\eta\eta} - I_{\xi\xi})\omega_\xi\omega_\eta. \end{bmatrix} \tag{I.6}$$

The model simulates the dynamics of a spacecraft in space with applied magnetic control approach. The ACS (air-bearing) SIMULINK® Model simulates the dynamics of the air-bearing model with applied magnetic control algorithm. It is not possible to easily create an artificial space-like environment on earth. So it was decided to design an air-bearing table setup to test the ACS control algorithm under conditions existing on the Earth's surface. The air-bearing platform design is

described in [Ref. 8]. This air-bearing platform consist of all necessary ACS components. It has its own power supply on board, all necessary sensors and actuators and the embedded ACS (control algorithm) SIMULINK® Model. Hardware-in-the-loop simulations of the ACS (control algorithm) SIMULINK® Model will be used to verify the correct implementation of the ACS control algorithm in SIMULINK®. Once the implementation is verified by comparing ACS (air-bearing) SIMULINK® Model simulation results and ACS (control algorithm) hardware-in-the-loop test results the ACS (NPSAT1) SIMULINK® Model and its simulation results will be verified.

One can estimate the validity of these hardware-in-the-loop simulation. It is a tool to test and verify the simulation of the magnetic control approach and to show possible problems, real world influences and other concerns to put that magnetic control approach into practice.

II. MAGNETIC MEASUREMENTS

This chapter deals with measurements to gain information and data of the actual Earth's magnetic field in the laboratory. These data are necessary to decide the location of the air-bearing table. They are also necessary to analyze the position data and control output data determined by the embedded ACS (control algorithm) SIMULINK® model on the air-bearing platform. The ACS (control algorithm) SIMULINK® Model uses data measured with the same device that is used to measure the Earth's magnetic field vectors **B** in these experiments. One may trace back occurring errors in future hardware-in-the-loop simulations to distorted magnetic data by comparing ACS (control algorithm) SIMULINK® Model data with data obtained from such magnetic measurements.

It is further intended to use an artificial magnetic environment for tests later on. This artificial magnetic environment is to simulate the changing magnetic field in a NPSAT1 orbit. There is only limited knowledge about handling a *Helmholtz coil* setup in the SSAG. It might be necessary to write custom software that produces the necessary input to the power-supplies of such a coil system. There are coil systems that null the Earth's magnetic field without further effort. The requested artificial magnetic field vector produced by such a coil system would be obtained directly from the magnetic data of a NPSAT1 orbit. Otherwise the requested artificial magnetic field vector produced by such a coil system has to be determined by vector addition of the Earth's magnetic field vector and the magnetic data of the orbit.

These measurements may also provide some information to improve the measurement setup and process and the analysis of measurements for the final installation of the air-bearing table at later time.

A. REQUIREMENT OF THE MAGNETIC ENVIRONMENT

The ACS (control algorithm) SIMULINK® Model determines the necessary magnetic dipole moment $\mathbf{m}$ to generate the control torque $\mathbf{T_c}$ to control the "attitude" of the air-bearing platform, based on comparison of measured and calculated environmental magnetic data. The magnetic control torque is applied with three magnetic torque rods aligned with the principle axes of the air-bearing platform. The measurement of environmental magnetic data is done by means of a magnetometer that is also aligned with the principle axes of the air-bearing platform [Ref. 5], [Ref. 7], [Ref. 8].

To minimize the influence of the magnetic and electronic hardware components (such as electro-magnetic effects of switching) on the measurements of the magnetometer, it is mounted away from these components and the magnetic control torque is applied in between two measurements. A disadvantage of this is that magnetic field data are measured at a different position from where the magnetic control torque is applied. The measurements of the laboratory magnetic field data have shown that there are deviations of the magnetic field vector at different positions. The air-bearing hardware-in-the-loop simulation will show what influence this has on attitude control.

The air-bearing platform is expected to rotate and/or oscillate on the air-bearing table. Depending on the rate of the air-bearing platform, the magnetometer will measure also magnetic field data at different positions with respect to the environmental magnetic field.

One can see from this and the principle of the ACS SIMULINK® Model that a homogeneous magnetic environment (with some tolerance) is necessary to support the function of the ACS SIMULINK® Model on the air-bearing platform. If the magnetic control torque (determined by and based on magnetic field data from a

particular position in the magnetic field) is applied at a different position with totally different magnetic conditions, the attitude control will not succeed in acquisition of requested air-bearing platform positions.

The air-bearing table is designed to keep all necessary magnetic components within a sphere of two feet in diameter, [Ref. 8]. This provides the use of a Helmholtz coil system that generates an homogeneous artificial magnetic field of that size. This is also a measure for how to choose measurement positions.

As long as the hardware-in-the-loop simulation is performed without this artificial magnetic field, the Earth's magnetic field has to be used. It has to be as homogeneous as possible at the temporary location of the air-bearing table. So the task is to find a position with minimal or small deviation relative to other positions.

B. MAGNETIC MEASUREMENT SETUP

1. Environment

The air-bearing table will be installed for first tests at a location that is used as storage and working room now. Therefore, some furniture and devices will be there during the tests such as lockers, desks, computers and monitors or two UPS (Un-interruptible Power Supplies). During the measurements these items were placed to match the arrangement during the tests. A part of the laboratory can be seen in Figure 4.

The SSAG building undergoes a reconstruction and a renewal of the electrical, water and heating system. The chosen laboratory is not under construction yet but some work may affect the magnetic environment in this area. A sequence of measurements was taken to establish if there is any influence on the magnetic field.

2. Hardware

Measurements were done by using the following devices:

Figure 1. Magnetic Measurement Hardware Setup.

- a Smart Digital Magnetometer HMR2300 from Honeywell,
- a Hand Held Display Module HMD5000 from Honeywell or
- a MICRON Pentium $166MHz$ Laptop and
- for mapping the magnetic field a non-magnetic tripod with a bubble-level.

The measurement hardware setup can be seen in Figure 1.

The magnetometer that is to be used on the air-bearing table, a Schonstedt Instrument CO. SAM-73C magnetometer, was replaced by the Smart Digital Magnetometer HMR2300 from Honeywell. The latest information about the SAM-73C status is a documentation sheet from 2nd March, 1998. As nobody could ensure that the magnetometer was not exposed to stronger electro-magnetic or permanent magnetic fields over longer periods of time and was not damaged, the decision was made to test the calibration before using it as measurement device again.

No information, no reference devices or data for comparison is available on how to test or recalibrate magnetometers of this type. Therefore, two Honeywell Smart Digital Magnetometers HMR2300 were bought offering the advantage is that they

use serial output making it easy to use it together with the ACS (control algorithm) SIMULINK® Model electronics on the air-bearing table and the handling of measured data. These Honeywell magnetometers offer additionally a set/reset function for realigning the permalloy magnetization. Some specifications of the Smart Digital Magnetometer HMR2300 are shown in Table XIV, Appendix A, [Ref. 9].

3. Software

The HMR2300 Smart Digital Magnetometer Demo Software, [Ref. 10], was used to operate the magnetometer. The MICRON Pentium166 Laptop was booted from a floppy disk in DOS mode to run this software. Later on the custom made LabVIEW^{TM} driver that is described in Section III.C was used for experiments .

This HMR2300 software provides different modes such as a compass mode or just a so called show-mode. This mode shows the measurement readings of the magnetometer split into $\xi,\ \eta,\ \zeta$ components respectively in the magnetometer body coordinate system and further the magnetic north direction in degrees. These readings correspond to `15,000` counts per 1Gauss (see also Table XV). These readings can be logged to a file, each measurement sequence to one file. These files contain the readings in four columns - one corresponding to each coordinate axis and one corresponding to the magnetic north direction in degrees. The columns are divided in a manner that MAPLE 7.0 is able to read those data without any formatting.

These logged reading values have to be converted to a unit of magnetic flux density (Tesla or Gauss). The conversion can be done by using the following relationship ([Ref. 9], [Ref. 10]):

$$\texttt{15,000}\ counts \mathrel{\hat{=}} 1\text{Gauss} \ \Rightarrow\ \mathrm{B}_{ij} \ =\ \frac{1\text{Gauss}}{15000 counts} \cdot (\text{reading value}), \qquad \text{(II.1)}$$

wherein $B_{ij},\ j \in \{\xi,\ \eta,\ \zeta\}$ is the corresponding component of the magnetic flux density vector at that measurement point. The conversion from the used unit Gauss of the *Gaussian measurement system* to *SI* units is ([Ref. 9], [Ref. 15]):

$$1\text{Tesla}\ (T)\ =\ 10000\text{Gauss}\ (G). \tag{II.2}$$

C. INFLUENCES ON MAGNETOMETER READINGS

General. Some tests were done to get to know influences on magnetic data measured with the HMR2300. The data sheet states values for accuracy and resolution. But it was wanted to get an impression of the meaning of those values.

The first experimental use of the magnetometer demonstrated the capabilities of that measurement device. It was found out that a simple ratchet would change the magnetometer readings within a distance of ca. $250mm$ of the device. E.g. a simple electronic watch in close proximity would change the readings only a little, but significantly.

The magnetometer can be used without any holder or frame but it has to be mounted for measurements on the air-bearing platform or on a tripod by means of some screws and mounting devices. The measurement setup for mapping the magnetic field in the laboratory can be seen in Figure 1 and Figure 4.

The mounting device is similar to that used on the the air-bearing. A first measurement was taken without any additional devices and with the magnetometer on a table fixed with two-sided tape. A second measurement was taken with added mounting devices, screws and bubble-level (see also Section II.D). Two samples of measurement were taken, each of 15 seconds duration for each configuration of the magnetometer setup.

Test for Equality of Mean of two Samples. One question was whether those mounting devices would have a significant influence on the magnetometer readings

ore not. Therefor were analyzed the first three columns (samples) of the data file, see Table XVIII, Appendix B. The statistic test was used to compare the location of mean of two independent samples, see [Ref. 11], [Ref. 13]. The principle of this test is described shortly in Paragraph II.D.3, Equations II.11 to II.16. The following assumptions were made:

- **distribution function** $\Phi(x)$:
  the distribution function of measured values of one sample is the Gaussian normal distribution, because some samples show a similar empiric distribution function, see II.D.3,
- **variance** σ^2:
  both samples have the *same known* variance σ^2, because the measurement range of the magnetometer was $\pm$1Gauss and the typical or minimal accuracy was given in %FS, see also Table XIV, Appendix A.

Hypothesis "H_0: The mean of two corresponding samples is equal ($\delta = d_0 = 0$)." was tested against "H_1: The mean of two corresponding samples is unequal ($\delta \neq d_0$)." With those assumptions mentioned above and σ as corresponding value to the minimum accuracy (that is the maximum allowance, see Table IV) Hypothesis H_1 was falsified with a confidence level of $(1 - \alpha) = 0.95$. The interpretation is that it is likely that there is no influence from this setup to magnetometer readings.

D. MAGNETIC FIELD

1. Measurement Grid / Coordinate Systems

The measurement of the magnetic field was taken aligned with the specified grid and coordinate system in Figure 2. The grid points are equally spaced along the chosen reference coordinate axes. The numbering of the grid points is lexicographical. The *reference coordinate system* was two dimensional and was extended later to a three dimensional coordinate system with the z axis pointing towards the earth. To

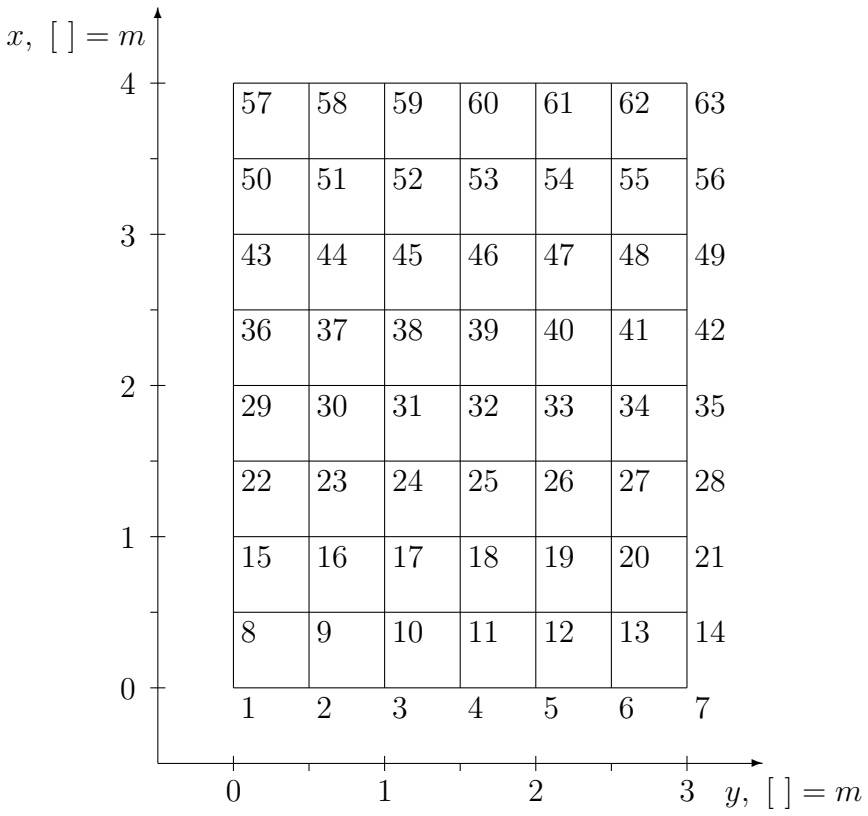


Figure 2. **B** Field Measurement Grid.

ease the alignment of this reference coordinate system, the axes were chosen to be parallel with the walls of the laboratory.

The air-bearing platform has an outer diameter of approximately $d_{max} \approx 500mm$. The HMR2300 magnetometer will be mounted on the outside of the air-bearing platform. The grid was chosen with a distance of $\Delta x = \Delta y = 500mm$ between two grid points .

The grid was drawn on the floor using a tape measure, chalk and a chalked line. At first it was square but then extended by two rows in the x direction to use all floor space.

It is necessary for correct interpretation of analyzed data to know the relation-

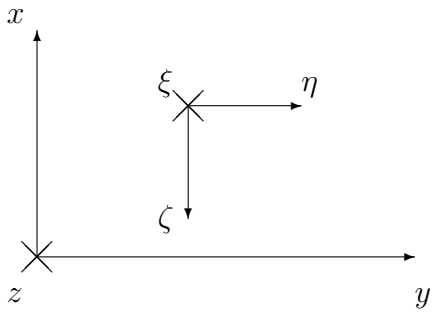


Figure 3. Grid and Magnetometer Coordinate System.

ship between the grid (reference) $[x,\ y,\ z]$ coordinate system and the magnetometer (body) $[\xi,\ \eta,\ \zeta]$ coordinate system. The magnetometer coordinate system is shown in the HMR2300 data sheet [Ref. 9] and additionally on a label on the device. The orientation of that coordinate system in the measurement setup (as well as on the air-bearing platform later on) does not conform to the orientation of axes of the grid (reference) coordinate system.

Figure 3 shows both the grid reference coordinate system and the magnetometer body coordinate system. Hence the coordinate transformation between those different systems is:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\zeta \\ \eta \\ \xi \end{bmatrix} \tag{II.3}$$

2. Measurement

The measurement of the magnetic field was done by means of the magnetometer HMR2300, the Laptop and the non-magnetic tripod. The magnetometer

Figure 4. Measuring the **B** Field.

was mounted on top of the non-magnetic tripod and was linked to the Laptop. The Laptop was standing on a non-magnetic laboratory cart and its position was changed during the measurement to keep a certain distance between the magnetometer and the Laptop (and its power supply). Figure 4 shows the measurement setup to measure the magnetic field in the laboratory.

The non-magnetic tripod with the magnetometer was positioned above one grid point and aligned with the grid. To ease the positioning of the tripod, a nut was hung beneath the telescope rod like a pendulum so that was only a small space between the nut and the floor. This way the center of the tripod could be positioned above the grid point. To ensure the alignment of the magnetometer body coordinate system with the reference (grid) coordinate system, one leg of the tripod was placed on a grid line. The position and alignment of the tripod and the magnetometer was fixed by these two points with respect to the grid. A bubble level was mounted on the tripod and facilitated the alignment with the horizontal plane.

To determine a volume of the **B** vector field the telescope rod of the tripod was used. It facilitated the measurement in planes at different heights. Its alignment and height with respect to the base tripod was checked by markings done with a

	Grid Point Index	*Height*
Plane 1	$1 \leq i \leq 63$	$38.5''$ $(977mm)$
Plane 2	$64 \leq i \leq 126$	$49.25''$ $(1250mm)$
Plane 3	$127 \leq i \leq 189$	$59.75''$ $(1517mm)$

Table II. Measurement Planes.

permanent marker pen. Measurements were taken in three different planes. Each height can bee seen in Table II.

After aligning the tripod and magnetometer with the grid and the horizontal plane were taken *ca.* $10s$ of readings from the magnetometer on each point. They were logged with the HMR2300 Demo software to data files. These files were named with the corresponding grid point number. As there are three planes of measurement points it was chosen to number them consecutively. This has the disadvantage of blurred information about the grid coordinates and the height in the file name. But with some index arithmetics the analysis of the data in these files is simplified.

3. Data Analysis

The measured field data were analyzed with MAPLE 7.0 from Waterloo Maple Inc. In Appendix B are shown and explained maple scripts that were used to analyze the data. In the following are presented some considerations with results and conclusions.

In Section II.A.3 is stated that the output data of the HMR2300 are given in the unit "counts". The conversions from this unit to "Gauss" or SI unit "Tesla" can be done with Equations II.1 and II.2. The measured magnetic field data were

analyzed without conversion from "counts" into a unit of magnetic flux density. The results of the analysis were converted into "Tesla" to allow better interpretation.

A coordinate transformation has to be implemented with unit conversions mentioned above when the magnetometer is used on the air-bearing platform.

a. Considerations about a Distribution Function.

In Section II.C was assumed that data measured with the HMR2300 have a Gaussian distribution function. No statistical test was made to test the fit of the assumed distribution function with actual distribution functions of measured samples.

In [Ref. 9] some operating specifications of the HMR2300 are given. In Table XIV, Appendix A are shown some important ones. The resulting accuracy of that device is given for scale ranges of ± 1Gauss and ± 2Gauss with an typical or minimum value in $\%FS$. These accuracy values were taken to determine a measure of deviation in a sample of data, all representing the same magnetic flux density vector component. The magnetic flux density vector field is continuous. The magnetic flux density vector $\mathbf{B}_i$ at a particular position i should be assumed to be constant for our purposes in the absence of artificial disturbances. Therefore an indefinite number of measurements $x_{ij},\ j \in \{\xi,\ \eta,\ \zeta\}$ can be taken at this location and should be measure *ideally* the same value every time if this is a true value. Hence all measured values at this position should be components of one population and the deviation is caused only by the measurement device. Artificial disturbances would lead to values from a different population because the $\mathbf{B}_i$ vector was changed and represents a different magnetic field. Therefore the specified accuracy values, multiplied with a corresponding scale range were interpreted as square root of the variance σ^2 of the resulting distribution of measured values of one sample $\mathbf{X}_{i,j}$:

$$\sigma = (scale\ range\ value)counts \cdot \frac{(accuracy\ value)\%}{100}. \tag{II.4}$$

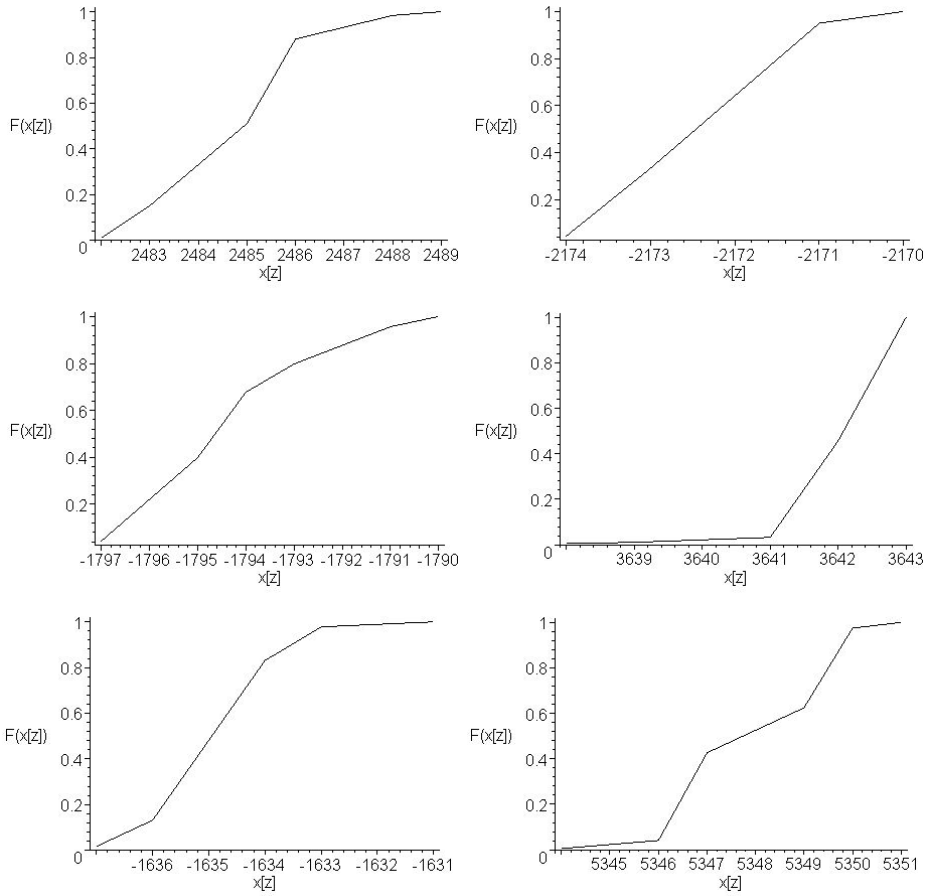
The measurements were taken from the continuous magnetic flux density vector field and it is commonly expected that most measurement devices deliver values as a function of a Gaussian distribution function, [Ref. 11]. Thus it was assumed that the distribution function of a sample $\mathbf{X}_{i,j}$ is the Gaussian distribution function. The empirical distribution function of certain samples was determined to get an impression of the quality of that assumption. The chosen samples are shown in Table III.

The empiric distribution functions of those samples was determined with following equation [Ref. 12], [Ref. 13]:

$$F(x(z)) = \frac{1}{N} \sum_{i=1}^{z} n_i, \tag{II.5}$$

with N as the number of measured values in a sample (dimension of the measurement vector), z as the number of classes in that sample (number of different readings) and n_i as the number of values in one class. The number of classes is in this case equal to the number of different reading values in one sample. It was decided to define classes in this way because most samples have three or fewer different reading values. This is also a reason for the considered samples shown in Table III. E.g. if there are only two different values in one sample the function would look like a linear function because a linear interpolation in between two points given by Equation II.5 was done. For smaller values than the smallest sample value and for higher values than the highest sample value is it difficult to define a reasonable graph of these functions.

Figure 5 shows examples for empirical distribution functions. The diagrams in the left column do look similar to the Gaussian distribution function that



top left	10.jan, #9, η sample	top right	17.jan, #45, ζ sample
mid. left	23.jan, #30, ζ sample	mid. right	10.jan, #9, ξ sample
bot. left	23.jan, #128, ζ sample	bot. right	23.jan, #7, ξ sample

Figure 5. Examples of Empiric Distribution Functions.

Date	*Grid Point* [1]	*Column*	*Number of Classes z* [2]
10.jan	9	2	6
	9	1	5
17.jan	45	3	4
	48	3	4
23.jan	7	1	6
	30	3	6
	100	1	6
	100	2	5
	128	3	5
	184	3	5
06.feb	30	2	10
	30	1	8
	118	1	5
	135	2	5

[1] See II.D.2 and Table II.

[2] The number of classes in this case equals the number of different reading values in the corresponding sample.

Table III. Samples for Determination of Empiric Distribution Functions.

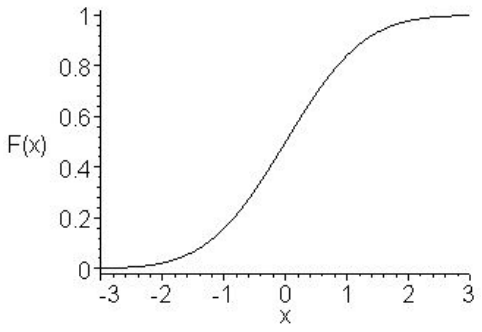


Figure 6. Gaussian Distribution Function.

can be seen in Figure 6. The diagrams in the right column do not look similar. The diagram on the top right is one example of an empirical distribution function determined from a sample with four classes. It is not that useful in considering such empiric distribution functions with four or fewer classes. The smaller the number of classes the more the distribution appears as a linear function (see above). That means that the sample would have an empiric distribution $F(x)$ like an equipartition distribution. But this is not a reasonable assumption in case of the genuine distribution function $\Phi(x)$ of the whole population.

The few samples that look similar to the Gaussian distribution function were taken as base for the assumption of a Gaussian distribution. The other ones are neglected, because most distribution functions tend to the Gaussian distribution function for infinite measured values x, *Central Limiting Value Theorem*, [Ref. 12], [Ref. 13], [Ref. 14]. Furthermore this is acceptable for this purpose and the Gaussian distribution can be assumed as an approximation.

From the *assumption of Gaussian distribution functions* in all measured samples follows for *one considered sample* $\mathbf{X}_{i,j}$:

$$\mu = E(\bar{x}(N)) = \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} x_i, \tag{II.6}$$

that means that the Expected Value (1st moment of corresponding distribution function) equals the arithmetical (empirical) mean of that sample for infinite numbers of measured values. $\bar{x}$ is an unbiased estimated value of μ, [Ref. 11], [Ref. 14]. A similar statement is possible for the empirical variance s^2:

$$\sigma^2 = E(s(N)^2) = \lim_{N\to\infty} \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2, \tag{II.7}$$

s^2 is the unbiased estimated value of σ^2, [Ref. 11], [Ref. 14]. These estimates $\bar{x}$ and s^2 are used for analyzing the measured **B** field data. The MAPLE 7.0 worksheet is shown and explained in Appendix B.

b. First B *Field Visualization.*

After deciding what parameter to use to describe one sample of measured data a first impression of what the laboratory **B** field was like was obtained. To visualize the data helped to determine a reasonable approach to choosing the position of the air-bearing table.

The first step to visualize the data was to reduce the amount of data for that visualization. This was done by using the describing parameters $\bar{x}_{i,j},\ s^2_{i,j}$ for one sample $\mathbf{X}_{i,j}$. One sample $\mathbf{X}_{i,j}$ is in this case one column of the three measured important ones (see Table XVIII in Appendix B) that is one column of data for each $\xi,\ \eta,\ \zeta$ **B** vector component on a particular grid point i, see II.D.1 and II.D.2. At the same time the confidence interval (*C.I.*) for each component $\mathrm{B}_{i,\xi},\ \mathrm{B}_{i,\eta},\ \mathrm{B}_{i,\zeta}$ of each magnetic flux density vector $\mathbf{B}_i$ could be determined if it might be necessary to do an error analysis of the hardware-in-the loop-simulation. C.I. can be determined with:

$$\bar{x}_{i,j} \pm u_{1-\frac{\alpha}{2}} \cdot \frac{s_{i,j}}{\sqrt{N_{i,j}}} \tag{II.8}$$

or rather

$$\bar{x}_{i,j} \pm t_{f_d,1-\frac{\alpha}{2}} \cdot \frac{s_{i,j}}{\sqrt{N_{i,j}}} \tag{II.9}$$

and the corresponding confidence level (*C.L.*) is given by:

$$(1-\alpha) \cdot 100\%, \tag{II.10}$$

Scale Range	*Accuracy*	*Deviation* σ
$\pm$1Gauss	typ. - 0.12%FS min. - 0.52%FS	18*counts* 78*counts*

Table IV. HMR2300 σ Values.

see [Ref. 11], [Ref. 13]. The t-quantile of Student's distribution was used because nearly all samples contain about $N_{i,j} \approx 180$ measured values. That means that the Student's distribution as function of $N_{i,j}$ has to be used instead of the Gaussian distribution. Thus it is necessary to differentiate between μ and $\bar{x}$ as well as between σ^2 and s^2, [Ref. 11], [Ref. 13]. That is why it was decided to be consistent and to use $\bar{x}_{i,j}$ and $s_{i,j}$ for determining C.I. of a sample $\mathbf{B}_{i,j}$.

Actually it appeared that $s_{i,j}$ derived from a sample was smaller than σ in Equation II.4. It was wanted to know if there have been short time disturbances during the measurements, e.g. from electrical devices used for the building reconstruction were investigated. The idea was to compare the standard deviation $s_{i,j}$ of a sample with the σ values from Equation II.4. These values can be seen in Table IV. If there had been a significant disturbance, it should have resulted in a bigger deviation value than determined from the minimal device accuracy. The values contained in one sample of shortly disturbed measurements should vary in a greater range than those in a sample with readings of an undisturbed $\mathbf{B}$ field vector. A disturbed measurement is in this case a $\mathbf{B}_i$ vector changed because of short time artificial disturbances. But if the actual deviation is smaller than the one specified it can not be decided whether the variation of values is caused by a disturbance that changes the $\mathbf{B}$ vector or just statistical influences and device properties.

Next the process of visualizing the measured data is described. The components of each vector $\mathbf{B}_i$ were determined as $\bar{x}_{i,j}$ of each sample $\mathbf{X}_{i,j}$, $1 \leq i \leq 189$ and $j \in \{\xi,\ \eta,\ \zeta\}$. They were scaled with respect to a grid frame. After scaling the

components $\bar{x}_{i,j}$ were combined in a vector $\mathbf{B}_i = [-\bar{x}_\zeta \ \bar{x}_\eta \ \bar{x}_\xi]^T$ (see Figure 3) and placed into a drawing grid. That visualization is convenient for qualitative information and *not* for quantitative analysis. But the intention was to get only qualitative information. The axes frame was included only to show the coordinate system and is not to scale. The Maple 7.0 work sheet is contained in Appendix B.

Just one example of visualized **B** field data is presented because there are only a few differences visible between the different vector field plots of 10th Jan, 17th Jan, 23 Jan and 06th Feb. The vector field plots of 10th Jan and 17th Jan contain only one measurement plane. Those measurements were taken when it was not clear if certain devices had to be removed from the laboratory and to check if the measurement setup was useful or had to be changed because of appreciable errors. The vector field plots of 23th Jan and 6th Feb contain three planes. The 06th Feb vector field plot can be seen in Figure 7.

As one can see in Figure 7, the center of the measured laboratory **B** field seems to be quite homogenous. But there appear deviation at the borders of that field. The rows in front ($x = 0 = const.$) show an increasing norm (or length) of the vectors from left to right ($0 \leq y \leq 300$), especially next to the right border. This effect can bee seen also at the right side of the volume ($y = 300 = const.$). A possible reason is that there are two Uninterrupted Power Supplies (UPS) next to this corner.

Conversely, the backside of the volume ($x = 400 = const.$) is showing stronger magnetic flux density at the left than at the right side. A solid reinforced concrete column is located at this corner .

Furthermore one can see that the vector direction in front is different from that in the center and at the backside of the volume. It seems that the vectors at the middle of the left side ($y = 0 = const.$) tend left compared to the other ones in the corresponding columns. Actually tend also the vectors at the backside backwards. In this area steel lockers were standing.

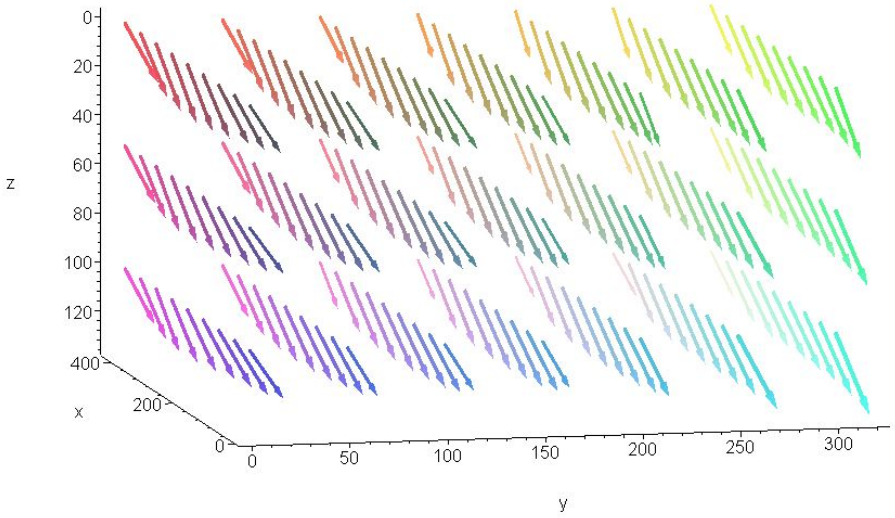


Figure 7. **B** Vector Field Plot I.

It is clear that there is more deviation in vector direction (or field direction) in the left than in the right field part. The deviation in vector norm (or field strength) is concentrated at the field borders. It seems from this Figure 7 that the middle right part of the volume center is the most homogenous part of the **B** field.

Looking at particular planes one can get a deeper understanding of what might be happening. The lowest plane of Figure 7 e.g. is shown in Figure 8.

c. Comparison of Repeated Measurements

One question concerning the four measurements of 10th Jan, 17th Jan, 23rd Jan and 06th Feb was whether they represent the same measured magnetic flux

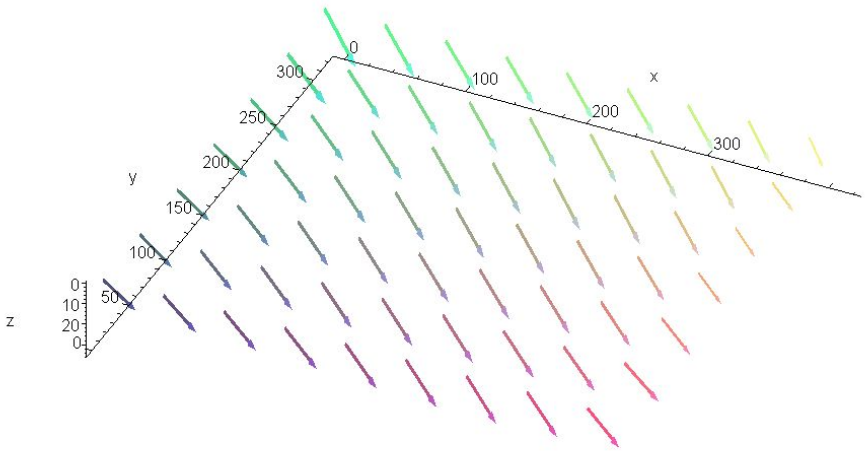


Figure 8. **B** Vector Field Plot II.

density field or not. If one has a look at Figure 29, Appendix B one can find a couple similarities between those repeated measurements.

The question if those repeated measurements represent the same magnetic field can be subject to the statistical test concerning the location of mean of two samples. This statistical test compares a test statistic (TS) as a function of the difference of both corresponding means with a quantile of the standardized Gaussian distribution function [Ref. 11], [Ref. 13].

Some prerequisites have to be met to get reasonable and useful results out of this test. The problem is that it could not be ensured that each repeated $\mathbf{B}_{i,j}$, $j \in \{\xi,\ \eta,\ \zeta\}$ sample at a particular grid point i is taken from the same population [Ref. 13]. This is a disadvantage of the measurement process that is described in II.D.2. One cannot ensure to avoid slightly different measurement positions from grid point to grid point and from measurement to measurement. This problem would not

have been occurred with more accurate alignment. So this is an artificial effect. It could be corrected if there would have been noticed each small misalignment. Hence the test serves only academical purposes. However running the test yields the expected result. Because of the problem mentioned above most samples do not belong to the same population. The test was done as described in [Ref. 11], a Maple 7.0 script is part of the data CD of this report.

A Gaussian distribution was assumed for repeated measurements at a particular grid point for the continuous magnetic flux density field. Both corresponding samples have the same known variance σ^2 (Section II.C) if they are taken from the same population (the genuine $\mathbf{B}$ field). σ^2 was taken because this should be the variance of the population (allowing deviation because of statistical errors) instead of the actual empirical variances s^2 determined from the actual standard deviation of both samples. If both samples represent the same component of the $\mathbf{B}_i$ vector, then the difference between their Expected Values should be zero:

$$\delta = \mu_1 - \mu_2, \tag{II.11}$$

with the actual difference

$$d_0 = 0. \tag{II.12}$$

The hypotheses of a two sided test are therefore:

$$\mathrm{H}_0 : \ \delta = d_0 \ vs. \ \mathrm{H}_1 : \ \delta \neq d_0. \tag{II.13}$$

The test statistic (TS) in the case of

$$\sigma^2 = \sigma_1^2 = \sigma_2^2 \tag{II.14}$$

is:

$$\mathrm{TS} = \frac{\bar{x}_1 - \bar{x}_2 - d_0}{\sigma\sqrt{\frac{1}{N_1} + \frac{1}{N_2}}}. \tag{II.15}$$

This test statistic has a standardized Gaussian distribution and is, therefore, compared with the u-quantile:

$$|TS| > u_{1-\frac{\alpha}{2}}. \tag{II.16}$$

If Equation II.16 is true, hypothesis H_0 is falsified.

As mentioned above, this test was conducted once and the result was as expected. H_0 was falsified in many cases. This result can be interpreted as caused by measurement setup and measurement process. And it is expected that this test would have led to different results without artificial contaminated data, [Ref. 13]. So was assumed that all measurements represent the same magnetic flux density field because of Figures 29, Appendix B and this should be reasonable.

d. Determining the Air-Bearing Table Position.

Several attempts were made to find an appropriate location for the air-bearing table. One attempt was to find a way that has used all information available resulting in the best fit to the mostly homogenous center section. It should avoid all distorted field vectors allowing to place the air-bearing at an arbitrary location in the inner magnetic field, that would be a location with few magnetic field deviations.

Principle Component Analysis.

In [Ref. 11] the approach of *principle component analysis* (PCA) and their application as "robuste Mittelung" (robust determination of an average value) is described. The intention of PCA is to find a *linear combination* of genuine measured data to represent the most common features of all that data. This linear combination is a so called *principle component*. The idea is to maximize the correlation of that principle component with all genuine measured data, [Ref. 11]. This could be interpreted as: each measured magnetic vectors is one sample and one has to look for a linear combination of those samples that contains most possible information. The resulting vector would have the maximum correlation with all measured vectors and represent the magnetic field.

Each vector is interpreted as a sample, each sample has three components. As explained in [Ref. 11] the next step is to standardize those samples, that means a transformation to a mean equal zero and a standard deviation equal one. The next step is to determine the correlation matrix and its eigenvalues and eigenvectors. Then one can obtain, after statistical testing and scaling, just one principle component.

But one can find at least two reasons force one to avoid this at first attractive approach. Both are based on the definition equation of the covariance and characteristics of correlation coefficients. The definition equation of the covariance is [Ref. 11], [Ref. 12], [Ref. 14]:

$$cov(X,Y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x-\mu_x) \cdot (y-\mu_y) \cdot \varphi(x,y) \cdot dxdy, \qquad \text{(II.17)}$$

with $X,\ Y$ as stochastic variables, their Expected Values $\mu_x,\ \mu_y$ and their distribution density function $\varphi(x,y)$. The equation for determining a correlation coefficient is [Ref. 11], [Ref. 12], [Ref. 13], [Ref. 14]:

$$\rho_{x,y} = \frac{cov(X,Y)}{\sigma_x \cdot \sigma_y}, \tag{II.18}$$

with

$$0 \leq |\rho| \leq 1. \tag{II.19}$$

The approach of PCI uses the matrix of empirical correlation coefficients. That matrix can be determined e.g. from the variance-covariance matrix [Ref. 11], [Ref. 12] because correlation coefficients are a function of covariance and variance, Equation II.18. The occurring problem is in this case the interpretation of vectors as samples.

A 3×3 artificial $[3, 1]$ vector field was considered. Seven vectors were equal with some superimposed small random numeric noise ε. Another vector was pointed in a different direction and another one has had a different norm (length). Unit vectors were chosen with each component $v_j = 1/\sqrt{3} + \varepsilon$, $j \in \{\xi,\ \eta,\ \zeta\}$. After subtracting the mean of each vector approximately zero vectors were obtained. The remaining components were the small random numeric noise ε. The characteristic of random or statistic numeric noise is that the correlation coefficients tend to zero. Therefore, seven independent correlation coefficients tending to zero were obtained. This is not a useful or acceptable result for this purpose. This weakness could be avoided by defining constraints to vector components so that there would remain a significant difference between random noise and centered component.

Another weakness is that correlation coefficients are a measure of linear dependence. If there is a high level of linear dependence between two samples their correlation coefficient tend whether to one or minus one. Actually is this the characteristic that would have sorted out all distorted directions. Unfortunately there is a high level of linear dependence between vectors of the same alignment but of different

norm. Because a set of aligned vectors can be obtained by multiplying a base vector with arbitrary coefficients.

Thus can be sorted out all misaligned vectors but not the ones of a different norm. Therefore, PCI was not used.

Median.

Another attempt was to use the median as a measure and to look for an air-bearing table position with a minimum of deviation with respect to this measure. The empirical median is defined as [Ref. 13], [Ref. 14]:

$$\tilde{x} \quad = \quad \begin{cases} x_k & , \ N = 2k-1 \\ (x_k + x_{k+1})/2 & , \ N = 2k. \end{cases} \tag{II.20}$$

The difference between mean and median can be described shortly as follows:

Mean $\bar{x}$ and **Standard Deviation** s. Both are one set of parameters to describe a distribution of a sample. They are used when the corresponding population's distribution is approximately or actually the Gaussian distribution. The mean is determined from all equally weighted values of the sample:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i, \tag{II.21}$$

so all information of that sample is used. One occurring problem or disadvantage is that extreme values (outliers or distorted measurements) have a great influence on the mean. If all measured values are from the same population the mean is a better estimate than the median. Its variation with repeated measurements is smaller than the variation of the median [Ref. 13], [Ref. 14]. The corresponding measure for the variation range of measured values is the standard deviation:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2}. \tag{II.22}$$

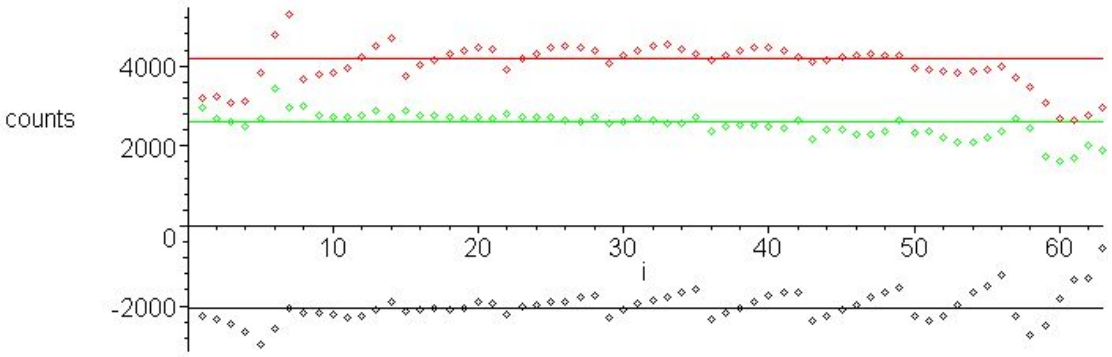
Median $\tilde{x}$ and **Interquartile Range** s_Q. Both are another set to describe a distribution of a sample. They are used if it is not sure that all measured values are from the same population. The median (see Equation II.20) uses only a part of the information of a sample. Its value divides the sample in two halves. Particularly extreme values do not have that much influence on the median than to the mean. Median and interquartile range are not that good estimates as mean and standard deviation but more robust e.g. with respect to outliers [Ref. 11], [Ref. 13], [Ref. 14]. The interquartile range can be determined from:

$$s_Q = \tilde{x}_{0.75} - \tilde{x}_{0.25}, \tag{II.23}$$

wherein x_p, $p \in \{0.25, 0.75\}$, is a p-quantile. A p-th part of all values in the sample is smaller or equal to this p-quantile and a $(1-p)$-th part of all values is equal or bigger than this value [Ref. 13], [Ref. 14].

It was tried to use the median as parameter to find an appropriate position within the field. Because of visible deviations at the borders of the measured field (Figure 7, Figure 8, Figure 9) it can not be assumed that all vectors are from the same population. As the same deviations appear in all plots they are not outliers. These deviations could be caused by superimposed magnetic fields. Hence not all samples have the same "source".

It is a little bit difficult to find the median of vectors. Information is presented in three components in three dimensional space. The question was how to determine the median vector. Component medians, taken of all ξ, η or ζ components, were combined to an artificial vector. This vector does not necessarily have a



ξ readings:	red diamonds,	ξ median:	red line
η readings:	green diamonds,	η median:	green line
ζ readings:	black diamonds,	ζ median:	black line

Figure 9. Median **B** Vector Components.

corresponding measured vector. But it should represent approximately most of the inner magnetic field vectors. The median vector was determined as:

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x}_\xi \\ \tilde{x}_\eta \\ \tilde{x}_\zeta \end{bmatrix}, \tag{II.24}$$

$$\tilde{x}_j = \begin{cases} x_{j,k} & , \ N_i = 2k-1 \\ (x_{j,k} + x_{j,k+1})/2 & , \ N_i = 2k, \end{cases} \tag{II.25}$$

and

$$j \in \{\xi, \ \eta, \ \zeta\}. \tag{II.26}$$

Figure 9 shows an example for the location of the median with respect to measured values of one measurement plane, $1 \leq i \leq 63$. The corresponding

samples were measured on Feb 6. Those diagrams contain the same information as the visualizations in Figures 7 and Figure 8. Information is presented in a more quantitative way and therefore does not give the impression of the field. Knowing that each row of the grid contains seven measurement points (see Figure 2) one can find the same effects as in the magnetic field vector plots. But one can see more clearly that the inner magnetic field is not that homogenous as it was hoped. There are only a few vectors that have approximately the same components (notice the scale). Actually it was desired to find a place where the grid point and its four surrounding grid neighbors would have similar magnetic flux density vectors. These median plots, like Figure 9, have shown that the median would not be one of the best solutions. If one takes a closer look at Figure 9 one can see that it is difficult to find vectors with three components close to the corresponding median values.

The following conclusion was reached that this median vector would offer measures to approximately describe a small part of the measured magnetic field. But because of the actual field inhomogeneity and the way this artificial median vector was determined the best position by using the median vector could not be found.

Difference Vectors.

The final attempt was to consider a particular grid point and its four neighbors. Because of such diagrams like Figure 9 it was decided to compare the vector at this particular grid point $\mathbf{B}_i$ with its surrounding vectors $\mathbf{B}_{i-7},\ \mathbf{B}_{i-1},\ \mathbf{B}_{i+1},\ \mathbf{B}_{i+7}$, see Figures 10 and 11. The $\mathbf{B}_i$ vector with the smallest amount of deviation from its neighbors would define the position of the air-bearing-table.

Comparing two vectors is not that difficult. One is defined as reference vector and the other one is compared to that reference. The challenge of this approach was a number of reference vectors (each grid point of the inner raster in Figure 11) and four neighbors have to be compared, see Figure 10. Furthermore there

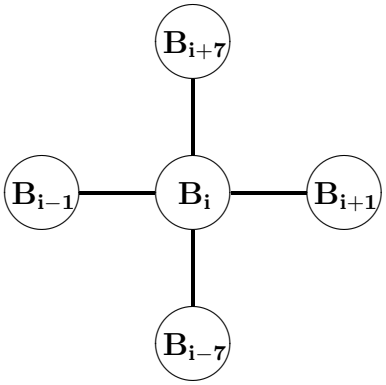


Figure 10. **B** Field Vectors to be compared.

are different deviations in field direction and field strength at each grid point. A single measurement that would accumulate this information about different field directions and field strength for each raster point has to be found. A single measurement should facilitate a clear comparison.

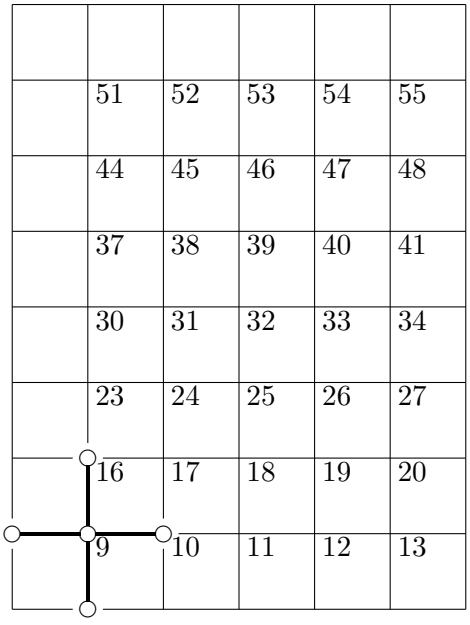


Figure 11. Raster for Comparing **B** Field Vectors.

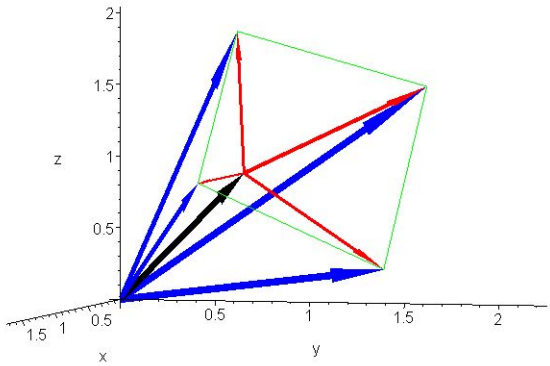


Figure 12. $\mathbf{B}_i$ and corresponding Difference Vectors $\mathbf{r}_{i,k}$.

By passing the "star" in Figure 11 across the grid five vectors like the four blue ones and the black one in Figure 12 may be obtained. This seems similar to solving a partial differential equation (e.g. Laplace's equation) with difference quotient approach. By subtracting the central vector from each other vector the corresponding difference vectors can be obtained:

$$\mathbf{r}_{i,k} = \mathbf{B}_k - \mathbf{B}_i,\ k \in \{i-7,\ i-1,\ i+1,\ i+7\}. \tag{II.27}$$

These difference vectors $\mathbf{r}_{i,k}$ are therefore vectors that point from the central vector to the corresponding neighbor, see Figure 12. Figure 12 is an artificial vector plot. The shown vectors are chosen to demonstrate the idea. $\mathbf{B}_i$ is drawn as black arrow. The surrounding vectors $\mathbf{B}_{i-7},\ \mathbf{B}_{i-1},\ \mathbf{B}_{i+1},\ \mathbf{B}_{i+7}$ are drawn as blue arrows. It is not necessary to specify them in the picture. The difference vectors $\mathbf{r}_{i,k},\ k \in \{i-$

$7,\ i-1,\ i+1,\ i+7\}$ are drawn as red arrows. The shown green lines determine the triangular areas spanned between two difference vectors $\mathbf{r}_{i,k}$.

These difference vectors are still not useful to determine a position of small deviation. Now one has to compare four vectors at a particular grid point with four vectors at each other grid point instead of five vectors. But these difference vectors contain information about deviation in field direction as well as in field strength. Now there has to be found a way to combine four vectors and two information into one measure. There has been figured out two possibilities to achieve this. Both are not perfect and have obvious weaknesses. But they are useful for this purpose.

The first idea, looking at Figure 12, was to use the sum of all triangular areas spanned between two adjacent difference vectors. These areas depend on the norm of both used difference vectors. The closer all $\mathbf{B}_k$ tops are to the top of $\mathbf{B}_i$ the smaller is the sum of all triangular areas. In the ideal case of five identical magnetic field vectors all areas vanish. The more they are spread out the bigger is the resulting area.

To make these resulting areas comparable they were normalized. Not absolute values but relative values need to be compared. A small resulting absolute area at the locations of the smaller field strength (vector norm) does not mean closer field vectors compared to locations of higher field strength and with the same relative deviation. A reasonable deviation measure to compare different field locations should be taken respectively to its actual location. Therefore the decision was made to normalize each difference vector $\mathbf{r}_{i,k}$ with the norm of its corresponding central vector $|\mathbf{B}_i|$. The equations to determine the little triangular areas and their sum follow:

$$\mathbf{r}^*{}_{i,k} = \frac{1}{|\mathbf{B}_i|} \cdot \mathbf{r}_{i,k} \tag{II.28}$$

normalizes each difference vector with the norm of its corresponding central vector.

$$a_{i,j,k} = \frac{|\mathbf{r^*}_{i,j} \times \mathbf{r^*}_{i,k}|}{2};\ j,\ k \in \{i-7,\ i-1,\ i+1,\ i+7\},\ j \neq k, \tag{II.29}$$

determines the triangular area between two adjacent difference vectors by means of the vector cross product. The norm of a resulting vector of a cross product can be interpreted as rectangular area spanned by its two base vectors [Ref. 14]:

$$|\mathbf{r^*}_{i,j} \times \mathbf{r^*}_{i,k}| = |\mathbf{r^*}_{i,j}| \cdot |\mathbf{r^*}_{i,k}| \cdot |\sin(\sphericalangle(\mathbf{r^*}_{i,j}, \mathbf{r^*}_{i,k}))|, \tag{II.30}$$

Indices $j,\ k$ are to be changed cyclical. The sum of all spanned triangular areas is:

$$A_i = \sum_{j,k} a_{i,j,k},\ \{j,\ k\} \in \{i-7,\ i-1,\ i+1,\ i+7\}. \tag{II.31}$$

The second idea was to use the sum of norm of corresponding normalized difference vectors:

$$L_i = \sum_{k} |\mathbf{r}^*_{i,k}|,\ k \in \{i-7,\ i-1,\ i+1,\ i+7\}. \tag{II.32}$$

One can see from Equations II.31 and II.30 that both approaches are based on norms of vectors. The difference is that one approach uses the sum but the other one uses a weighted product. While one approach uses only information given by vectors the other one uses additionally information about the relative position between these vectors. Both approaches have disadvantages. But these disadvantages will not lead to totally wrong conclusions. No better solutions for the problem of comparing couples of vectors was found.

Some disadvantages of both approaches follows. The use of the *sum of norm* of corresponding vectors is easier for calculation. Multiplying the *sum of norms*

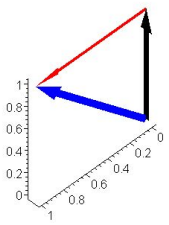


Figure 13. One Critical Analysis Case.

of these normalized difference vectors with the norm that is used to normalize them results in a value that has the same unit like the magnetic flux density field vectors: Tesla. So it seems that one has a measure that is easily interpretable. Conversely the *sum of area* approach is not that easy to handle for interpretation.

There are more disadvantages. Consider five vectors, four with the same length and pointing in the same direction and one central vector with an arbitrary length and pointing in a different direction, e.g. Figure 13. All difference vectors pointing from this central vector to the other four ones have therefore the same length and direction. So the deviation measure *sum of norms* would result in four times the norm of the normalized difference vector. The deviation measure *sum of areas* would result in zero deviation because no areas are spanned.

Consider five vectors in this way that they result in five normalized difference vectors located in one plane, with equal length and pointing in directions $90 degrees$ to each other, e.g. Figure 14. The deviation measure *sum of norms* would again result in four times the norm of the normalized difference vectors. The deviation measure *sum of areas* would result in a number different from zero. These examples give a short view on the number of possible occurring problems. But their occurrence is hypothetical and these are theoretical considerations. If one has a look at Figure 9

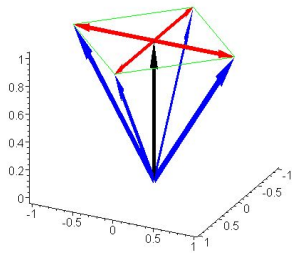


Figure 14. Another Critical Analysis Case.

and Figure 29, Appendix B, one can see that it is unlikely to find five adjacent magnetic flux density field vectors that share the same plane.

It was decided to use the *sum of areas* approach. One can see in Equation II.30 that the sinus of the angle between two adjacent normalized difference vectors has influence. This angle is a measure for the relative position of these two vectors to each other. Since cases like the one shown in Figure 13 are unlikely this *sum of areas* approach was tested with real magnetic field data. The results are shown in Table V. The comparison with Figures 29 was encouraging.

A mathematical proof of the functionality of this *sum of areas* approach was not attempted. It is obvious that there would be only few use of doing this, because of the shown examples. Furthermore because of the characteristics of the measured magnetic field (inhomogeneity) and characteristics of the data (statistical errors) is expected that this approach serves its purpose although it is not approved mathematically. At least it was the best tool that was available and has lead to reasonable results.

The application of this *sum of areas* approach and the results of it are described in the next Paragraph II.D.4.

4. Results

In Paragraph II.D.3 was assumed that all repeated measurements (10th Jan, 17th Jan, 23rd Jan, 06th Feb) represent the same magnetic flux density field. This assumption together with the *sum of areas* approach to determine the position of the air-bearing table was used.

The *sum of areas* approach was implemented to search for the five smallest spanned areas in one measurement plane, see Appendix B. The number of five was chosen instead of the single minimum value because they do not vary that much and give therefore a better impression of the results and possible relationships.

The results have been encouraging because no multiple values have appeared and there are a few grid points that have a higher frequency of occurrence. Another fact is that these selected or filtered grid points are located mostly in a particular area of the measurement grid, see Table V and Table VI together with Figure 2. The frequency of occurrence of particular grid points can be seen in Table VI. The frequency of occurrence was taken based on eight measurements each with five values. All planes were considered equally. Even if one considers only corresponding planes the results do not change that much.

The grid point that has the largest frequency of occurrence is grid point 32. This is the central point of the grid. Other grid points with a large frequency of occurrence are located next to this central point, see Table V together with Figure 2. Grid point 32 was chosen to install the air-bearing table on.

The values at grid point 32 measured at 06th Feb are presented as resulting magnetic flux density field at the position of the installed air-bearing table. These magnetic flux density vectors are shown in Table VII. The surrounding magnetic flux density vectors can be found in Table VIII. In Table VII is stated also the estimated value of the undisturbed Earth's magnetic field, [Ref. 4].

Data Set	*Grid Points*			*Data Set*	*Grid Points*		
10th Jan	25 26 27 32 37			17th Jan	26 27 31 32 33		
23rd Jan, 1st plane	25 26 27 31 32	2nd plane	25 26 32 33 34	06th Feb, 1st plane	24 25 26 32 33	2nd plane	20 27 32 33 34
3rd plane	19 27 32 33 34			3rd plane	18 19 33 34 38		

Table V. Filtered Grid Points.

Grid Point Nr.	*Frequency of Occurrence*[(1)]
25	4/40
26	5/40
27	5/40
32	7/40
33	6/40
34	3/40

[(1)] All to 40/40 missing values are dispersed on the other grid points shown in Table V.

Table VI. Frequencies of Occurrence of Grid Points.

Height	i	$\mathbf{B}_i^T$, [ ] = T	C.I.$_{\alpha=0.05}$ ($\mathbf{B}_i^T$, [ ] = T)
$977mm$	32	[.123511e-4, .175909e-4, .300628e-4] $\|\ \| = .369562e-4$	[[.123502e-4, .175905e-4, .300623e-4], [.123521e-4, .175914e-4, .300633e-4]]
$1250mm$	95	[.125828e-4, .170747e-4, .300667e-4] $\|\ \| = .367951e-4$	[[.125823e-4, .170743e-4, .300667e-4], [.125833e-4, .170751e-4, .300667e-4]]
$1517mm$	158	[.122752e-4, .169865e-4, .303982e-4] $\|\ \| = .369225e-4$	[[.122747e-4, .169861e-4, .303977e-4], [.122757e-4, .169869e-4, .303987e-4]]
IGRF **B** *Field, lat=36.4069deg, lon=122.134deg, h=17m*			
$\|\mathbf{B}\| = 0.515315$T			

Table VII. **B** Field at Air-bearing Table Position.

One can see that the measured $|\mathbf{B}_i|$ values have the same order as the IGRF reference value but are significantly smaller than this reference. This might be a result of artificial magnetic fields superimposed with earths magnetic field. The SSAG building is built with reinforced concrete and there are lots of visible and hidden cable ducts. Since the real magnetic environmental conditions are unknown in this area this assumption can be made and is reasonable. Components are not compared since there are uncertainties concerning the measurement directions. This is mentioned in Paragraph II.D.3.c.

E. MAGNETIC MEASUREMENT CONCLUSIONS

The magnetic measurement setup (Section II.B) and the measurement process (Paragraphs II.D.1, II.D.2 are useful and have served their purpose to get to know the actual magnetic field. The measurement setup is simple and can be used by anybody without any problems.

A disadvantage of the measurement process is the alignment of the measurement setup with the chosen reference coordinate system. Since the alignment is done

i	$\mathbf{B}_i^T$, [ ] = T	C.I.$_{\alpha=0.05}$ ($\mathbf{B}_i^T$), [ ] = T
25	[.126921e-4, .180758e-4, .296538e-4] $\|\ \| = .369753e-4$	[[.126918e-4, .180751e-4, .296535e-4], [.126925e-4, .180765e-4, .296541e-4]]
31	[.130421e-4, .179265e-4, .292727e-4] $\|\ \| = .367198e-4$	[[.130416e-4, .179262e-4, .292725e-4], [.130425e-4, .179269e-4, .292730e-4]]
33	[.119225e-4, .171299e-4, .301736e-4] $\|\ \| = .366882e-4$	[[.119219e-4, .171294e-4, .301733e-4], [.119230e-4, .171303e-4, .301739e-4]]
39	[.125333e-4, .167979e-4, .297119e-4] $\|\ \| = .363600e-4$	[[.125333e-4, .167975e-4, .297114e-4], [.125333e-4, .167984e-4, .297123e-4]]

Table VIII. Air-bearing Table Surrounding **B** Field Vectors (Lowest Plane).

manually just with simple optical references its accuracy is adequate. It was obtained a good overview of how far data analysis of these data is reasonable and necessary. Another disadvantage of the measurement process is the time that has to be spend measuring a volume of the **B** field. So one or two measurements should be enough with respect to the disadvantage mentioned above.

The measurement setup does likely not affect to the measurements, see Section II.C. However when the air-bearing table is set up, a comparison between previously measured magnetic environment and measurements taken from the magnetometer mounted on the air-bearing table could be useful for error search/handling and data analysis.

The extended analysis of the measured data has shown that the common approach of describing a measured sample with mean and standard deviation is reasonable, see Paragraph II.D.3. Since the measurement setup does not ensure that repeated measurements result in comparable samples, mean, standard deviation (and if necessary confidence interval) describe just one sample with respect to the actual configuration. Averaging over repeated measurements could be done by using e.g.

the median. This is considered to be not necessary and only reasonable for some more measurements.

The measurement process has led to reproducible results (with respect to its obvious inaccuracy). The repeated systematical effect that can be seen in Figure 29, Appendix B is one example for that. The actual task has been to determine a position with only a small amount of magnetic field deviation. It has been neglected to describe this effect with e.g. a fit, since there is no actual necessity.

The chosen approach to determine a position of little magnetic deviation has been successfully although it may be ambiguously, see Paragraph II.D.3. The obtained results (Paragraph II.D.4) have matched with expectations based on the visualizations of the measured data in Figure 29. The obtained results does not match with an IGRF model reference value. But they are from same order and the difference between reference value (based on a model of the Earth's natural magnetic field) may be caused by superimposed artificial magnetic fields.

III. HMR2300 MAGNETOMETER DRIVER

This chapter describes the development of custom made driver programs to communicate with the HMR2300 magnetometer. The SSAG decided to use the HMR2300 magnetometer from Honeywell instead of the Schonstedt Instrument CO. SAM-73C magnetometer. That decision has made the use of additional analog-digital conversion hardware and software unnecessary. Therefore some hardware components could be removed from the air-bearing platform. The HMR2300 facilitates serial communication via COM ports. Some basics of serial communication can be found in [Ref. 24].

Implementing SIMULINK® supported xPC Target serial magnetometer driver was expected to be easier with information and experience obtained from implementing a serial LabVIEW^{TM} magnetometer driver. This was done without serious problems and this approach has served its purpose as far as it could.

Work on the SIMULINK® supported xPC Target magnetometer driver is not completed. MATLAB SIMULINK® offers a couple of facilities for serial communication. Progress is made only with time consuming experiments that often can be characterized as "trial and error". Questions and problems have not been covered in available documentations. Even Mathworks technical support contacted via telephone or e-mail could often only give unsatisfactory information. The major attempts developing an xPC/SIMULINK® driver are presented.

A. MAGNETOMETER HMR2300

The technical specifications that are necessary for measurements and analysis are presented in Table XIV, Appendix A. Now some specifications of that device are presented that are necessary to communicate with this device. These specifications are stated also in [Ref. 9].

The HMR2300 has an internal analog digital converter. The data are serially output using the RS232 or RS485 standard for serial input to most personal computers. A command set is provided to configure the device and initiate measurements. These commands can be typed in through a standard keyboard while running any communication software such as Terminal in Windows®.

The output data format can either be 16 bit signed binary (sign + 15 bits) or binary coded decimal ASCII. In Table XV, Appendix A, are shown the BCD ASCII outputs for magnetic field values between ±2 Gauss. This format is easier for direct interpretation by the user. However, it may be necessary to use binary format for computer applications. Some examples for binary values between ±2 Gauss and both ASCII and BINARY output formats are also shown in Table XV and [Ref. 9].

An additional advantage in using the HMR2300 is that it offers a set/reset function. This function can be used to realign the permalloy magnetization, cancel out any temperature drift effects and yield the maximum output sensitivity for magnetic sensing. One has to be careful because only in the "set mode" the directions of sensitive axes correspond to the coordinate system that is shown on the package label and in technical drawings. In the "reset mode", sensitive field directions are opposite to those shown.

Table XVI, Appendix A, contains some important command inputs. Table XVII, Appendix A, shows some important time values. These values can be necessary if the command inputs are sent by a computer application and are not typed into a keyboard. If output commands are sent to quickly to the device it may not respond as expected.

B. CONSIDERATIONS ABOUT DRIVER ARCHITECTURE

The approach to communicate with the HMR2300 was to develop and build at first an instrument driver with *National Instruments LabVIEW*^{TM}. This software package is designed to control measurement devices besides other applications such as experiment monitoring, data acquisition, controlling etc. It should be easier to design and build a MATLAB SIMULINK® instrument driver with information and experience obtained in this way. This MATLAB SIMULINK® instrument driver is expected to be implemented within the software setup of the ACS SIMULINK® hardware-in-the-loop simulations. It was expected to be more difficult to design only the MATLAB SIMULINK® instrument driver without experience in communicating with the measurement device via other software than the HMR2300 Demo software, [Ref. 10]. Indeed, the process of building the LabVIEW^{TM} instrument driver has shown some possible problem sources in communicating with the HMR2300 magnetometer.

National Instruments (*NI*) LabVIEW^{TM} is designed for using PCs as measurement device controllers. So it offers documentations and online help as well as tutorials on the NI web site that deal with a wide spectrum of applications, see [Ref. 16]. One of the topics is developing a LabVIEW^{TM} instrument driver. This topic deals not only with software specific information but also with general information.

1. External and Internal Design Model

Figure 15 is taken from [Ref. 17]. This document was created to support the development of instrument drivers that control programmable instruments. Its intention is primarily to establish standards for driver structure, device management, instrument I/O and error reporting. It exists a variety of instrument drivers and these standards should facilitate the direct use of available drivers to unexperienced users. It is also a good introduction to a process for developing useful instrument drivers.

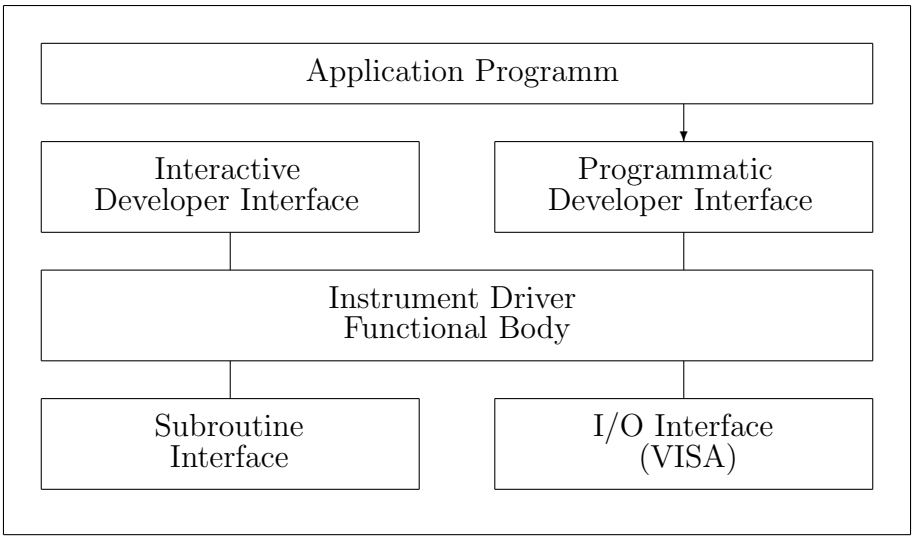


Figure 15. LabVIEW Instrument Driver External Design Model.

A useful instrument driver should facilitate an interactive call from the user or a call from a higher-level application software or both. The interactive developer interface in Figure 15 assists in understanding the functions of the driver and how to use the driver. This would be the front panel. The functional body is the code for the instrument driver. The I/O interface is the mechanism through which the driver communicates with the instrument hardware. VISA is a *Virtual Instrument Software Architecture*.

The subroutine interface is the mechanism through which the driver can call supporting applications that are needed to accomplish a task, such as error messaging [Ref. 17].

A possible functional body of a driver software is shown in Figure 16 [Ref. 17]. This model offers a possible structure that is based on experience. All available instrument drivers on the National Instruments web site conform with this structure. If the end user has understood this model and its background he should be able to use any LabVIEW^{TM} instrument driver that conforms to it. Furthermore, one can

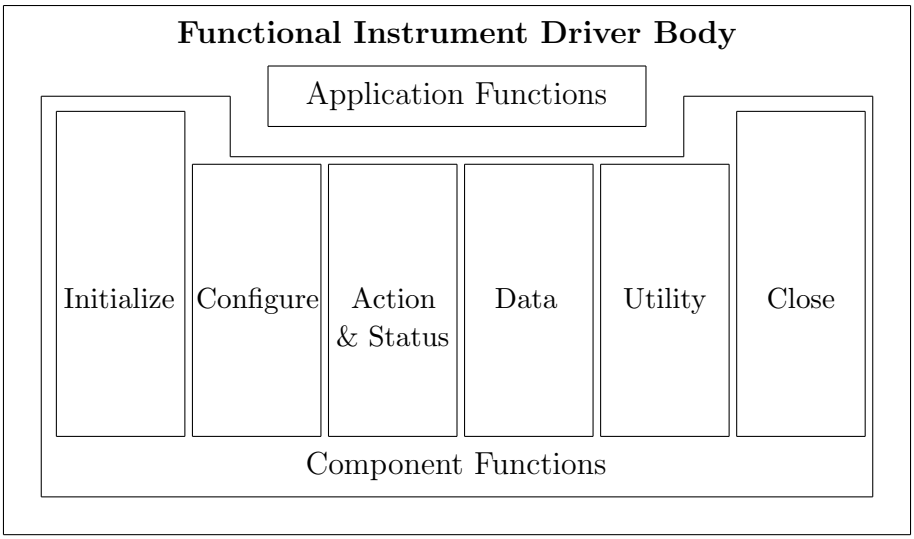


Figure 16. Functional Body of an Instrument Driver.

customize easily these instrument drivers to fit the requirements. This model does not require a specific software. It shows a rather reasonable approach for instrument driver development.

The first attempt to communicate with the HMR2300 magnetometer is a LabVIEW^{TM} instrument driver. Therefor was decided to use this model as a guideline. Additionally the LabVIEW^{TM} help topic "Developing a Simple Driver" [Ref. 18] was used because the necessity was only to create an application that has facilitated measurements and not access every feature of the magnetometer. A useful instrument driver is one that does what the user needs.

The *Application functions* in Figure 16 are the most advanced functions in the internal design. They call other appropriate functions and perform the most commonly used instrument configurations and measurements. Other higher-level applications or the end user interaction with the device are accomplished by using such application functions.

The *Component functions* are functions called by application functions. They send commands and receive device responses. Or they support in a different manner the use of the device, such as error messaging. These component functions can be divided in different groups, each with an own task. These groups are [Ref. 17]:

- *Initialize* functions,
- *Configuration* functions,
- *Action* and *Status* functions,
- *Data* functions,
- *Utility* functions,
- *Close* functions.

Initialize. This is the first function that calls the device. Its task is to establish communication with the instrument.

Configuration. These functions prepare the instrument to fulfill its current task. For instances they send commands and place the device in the requested state to perform measurements or stimulate a system.

Action/Status. Action functions cause the instrument to initiate or terminate measurement or test operations. They can also arm a triggering system or stimulate a second system. Action functions do not change the state of the device as configuration functions do. Status functions obtain the current status of the instrument or an action.

Data. The data transfer to or from the device is done with data functions.

Utility. These functions facilitate a wide spectrum of usable features like reset, calibration, storing and recalling instrument configurations, etc.

Close. The close function terminates the communication with the instrument and deallocates system resources.

2. Instrument Driver Structure

A useful simple instrument driver does not need to have all of these functions. The current application of the device defines the necessary function.

The structure of an instrument driver is based on the current application and on the specifications of the device. If one has some experience in using the device with other applications, one knows usually what steps have to be taken to get a desired result. The user manual of the device includes a section dealing with command sets and other programming specifications. A useful and reasonable structure of the instrument driver can be determined from the desired application, the command hierarchy and experience.

Based on the external and internal design model and with some knowledge about specifications of the device, whether a simple queue structure or a modular structure can be developed. If the instrument properties allow it and the task requests it, the structure could be simply: establish communication - take measurement - terminate communication or *open - action - data - close*. Therefore it is unnecessary to build an advanced instrument driver with modules for initializing, configuring, action/status etc. But if it is necessary to do so, the modular structure helps in keeping an overview and allows to change or extend the program more easily. The different modules would be connected but could be edited or replaced without affecting directly the structures of other modules. The modular design facilitates also an easier understanding of the function of an instrument driver because each module performs a logical task or function.

The structure mostly defines the execution order of modules or functions. But if the device is left after completing a function in the wrong state and is not ready to execute the next function, the command may be ignored or cause an error message. Additional timing problems occur if the data that are to be read are not yet available or if the next command is sent when the device is still busy executing the previously

sent command. Possible solutions are to integrate timers, use timeout functions or use status information. They have to be included in the structure. Information about necessary time delays / response times should be included in the user's manual of the device or need to be determined in a different manner, [Ref. 17], [Ref. 18].

3. Requirements for a Custom-made HMR2300 Driver

In Section III.B is stated that the development of a HMR2300 magnetometer LabVIEW^{TM} driver was only to experiment with the device and obtain experience as well as information about possible problem sources. After that it was expected to be easier to implement a HMR2300 magnetometer MATLAB SIMULINK® driver. In the last Paragraph, III.B.2, are described some information and considerations about a general instrument driver structure. Some considerations that influences the design and structure of the custom made HMR2300 magnetometer driver are presented in this paragraph.

In Section I.C is given a short overview of intentions of the ACS SIMULINK® hardware-in-the-loop simulations. The verification of the ACS SIMULINK® Model is to be done with a real-time application of the MATLAB SIMULINK® ACS control algorithm and real hardware input (from a magnetometer) and output (to the magnetic torque rods). The real-time application is a SIMULINK® xPC Target application built with *Real-Time Workshop*® from Mathworks, running with MATLAB®. This approach verifies directly the SIMULINK® implementation of the ACS control algorithm and simultaneously the ACS (NPSAT1) SIMULINK® Model, see also [Ref. 7].

From the SIMULINK® implementation arise some requirements for the HMR2300 magnetometer driver. It has to be compatible with the embedded xPC Target real-time application of the ACS (control algorithm) SIMULINK® Model that handles the magnetometer data. Real-Time Workshop® compiles any SIMULINK® model that conforms to some requirements such as only discrete states etc. into an

xPC Target application ([Ref. 7], [Ref. 20], [Ref. 21]). This application is downloaded to the so called *target PC*. This target PC is booted with xPC Target and runs only the xPC Target application in real-time. It is linked to the so called *host PC* that runs the MATLAB® environment. As long as the xPC Target application is executed on the target PC the user has only few possibilities to change properties or interact with the xPC Target application. These possibilities like Signal Tracing, Signal Logging and Parameter Tuning are described in [Ref. 20], [Ref. 21].

The process of creating and running an xPC Target application means for the custom made HMR2300 magnetometer driver that whether the whole driver is implemented within the ACS control algorithm SIMULINK® model or has to be running on an extra board. But the communication interfaces for communication between xPC Target application and driver have to be implemented within the SIMULINK® model. The higher-level xPC Target application calls the HMR2300 magnetometer driver. An HMR2300 driver implementation within the ACS (control algorithm) SIMULINK® model means use of SIMULINK® I/O support. The second solution of an extra driver board was cancelled out because of additional time and effort in board development and verification.

One of the settings that have to be done before compilation a SIMULINK® model is to specify sample times of SIMULINK® blocks within the xPC Target application. These sample times specifies the frequency the xPC Target application executes these blocks and updates their states. This means to the HMR2300 magnetometer driver to fit into the ACS algorithm magnetometer sample cycle. Whether the implemented SIMULINK® HMR2300 driver has to synchronize the device with the SIMULINK® implementation of the ACS control algorithm if the device is not sending data continuously. Or the SIMULINK® implemented ACS control algorithm has to call the driver sampled to cause single outputs. Continuous output means that whether the HMR2300 magnetometer driver or the SIMULINK® implemented ACS control algorithm have to grab just one value out of the continuous data stream and

dump the rest. Single output would be easier to handle because just one measurement sample is taken. Work would be done by the HMR2300. Therefore I consider continuous output as not useful.

The ACS control algorithm in its current implementation requires environmental magnetic flux density field data every two seconds. So it is not necessary to take continuous measurements, see output (Table XV) and timing specifications (Table XVII). Further the ACS algorithm needs only magnetic flux density vector information and nothing else. Therefor the HMR2300 magnetometer driver has only to establish communication with the device, set it into the right measurement mode (single output command) and cause output. Since user defined settings can be stored in the EEPROM of that device (see [Ref. 9]) the driver structure does not require a configuration tool. It could be just that simple as: establish connection - cause output.

It was decided to use the general structure of a custom made HMR2300 magnetometer driver that is shown in Figure 17. The underlined properties in box *Configure* are the preferred settings. This box is intended to be implemented to be used for experimental purposes. The Binary data format seems easier to handle than the ASCII format, see the weird "comma rules" in Table XV, [Ref. 9]. The decision single vs. continuous output depends on software features. If the software facilitates to grab just one value out of a continuous data stream and dump the rest, both encapsulated boxes *Action* and *Data* are to be executed only once. *Action* to cause continuous output and *Data* to read in continuously. Otherwise they have to be executed repeatedly to provide single measurements, refer to the command sets in Table XVI, Appendix A. One can see in Figure 17 that not all functions shown in Figure 16 are used. But the general HMR2300 driver structure covers most of them. The implementation of each of these functions will depend again on software specific features.

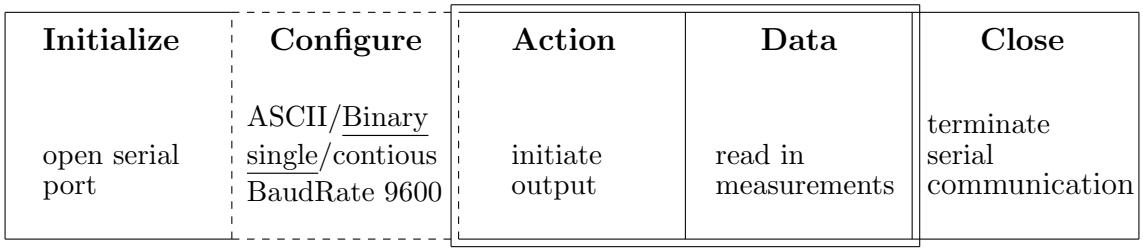


Figure 17. General HMR2300 Magnetometer Driver Structure.

C. NI LABVIEW^{TM} DRIVER

The knowledge about design models and driver structures in Paragraph III.B.2 and considerations in Paragraph III.B.3 have been the base for building the LabVIEW^{TM} magnetometer driver on. As sources [Ref. 17] and [Ref. 18] are NI sources, they have provided also some information about software specific features.

1. LabVIEW^{TM} Magnetometer Driver Structure

The requested task of the magnetometer has been to provide the ACS SIMULINK® Model with data of the magnetic environment. The ACS SIMULINK® Model was designed to use single magnetic flux density field (**B**) vector samples to determine the current configuration of the spacecraft/air-bearing platform and to determine the necessary magnetic control torque [Ref. 5], [Ref. 7], Paragraph I.B.3. These samples are requested after pulsing the magnetic torque rods and their magnetic decay, but before the next pulsing.

The measurements of the **B** field, see Chapter II, and the user's manual of the HMR2300, [Ref. 9], have shown how to use the HMR2300 together with the Honeywell HMR2300 Demo software, [Ref. 10]. The manual has provided also a command set and timing specifications, see Table XVI and Table XVII, Appendix A.

It was decided to build the HMR2300 magnetometer LabVIEW^{TM} driver with an option to act in any requested sample cycle if possible. But intention was to do

the sampling of the magnetometer with the ACS (control algorithm) SIMULINK® Model to avoid the task of synchronization two independent systems. The intention was to control the device with the xPC Target application to keep things simple. LabVIEW^{TM} has provided some tools to sample the magnetometer with a LabVIEW^{TM} magnetometer driver.

A single queue structure was chosen for implementing the HMR2300 driver, without building different modules although the final appearance may look different. The driver application starts with **initialization** of the serial communication to the HMR2300 that is connected to a serial COM port of the computer. The COM port address is passed in this phase and the HMR2300 send buffer is checked additionally for unexpected outputs (see Paragraph III.C.2). The **configuration** phase follows. The device is put into its default status (see Table XVI). This is the most convenient configuration for the requested experimental task. Additionally a possibility to change the data format from ASCII to Binary and to store these settings is implemented. This output format could be more convenient for converting output from values with unit "*counts*" to Tesla (or Gauss) values because of the inconvenient "comma rules", see Table XV. After the HMR2300 is configured, the HMR2300 is commanded within the **action** phase to measure the components of the **B** field. Those **data** are read from the device. In this driver application they are not logged to a file. They are just displayed. When the data package is read, the **close** phase terminates the communication with the HMR2300 and warnings or errors are displayed.

2. Implementation

The following tools of the LabVIEW^{TM} library were used in different numbers:

- VISA Open - to establish the connection to the HMR2300,
- Property Node - to specify the number of bytes currently available at the serial port,

- VISA Read - to read the device response after an output causing command or to "empty" the device buffer,
- VISA Write - to send command strings (see Table XV),
- Concatenate String - to concatenate the command and the Carriage Return Constant into one output string,
- Wait (ms) - to include appropriate time delays between send commands and responses to be read,
- VISA Close - to terminate correctly the connection to the HMR2300,
- General Error Handler - for debugging,
- Sequence Structure - to define explicitly an execution order,
- While Loop - to facilitate sampled measurements.

If one takes a closer look at the command set of the HMR2300, the use of the Property Node and the while loop may not seem to be necessary. Actually the while loop is the element that replaces the command for reading outputs at a device-controlled sample rate (`*ddC <cr>`, see Table XVI, Appendix A). Its implementation was necessary because the VISA Read function does require a specified number of bytes that are to be read from the device. The while loop can be executed until it is switched off manually and the sample rate was defined by a Wait (ms) tool that has controlled the execution rate of the while loop.

The Property Node was included after some testing. After stopping the execution of the driver application without using the VISA Close function and starting the application again, the ξ, η, ζ output readings changed places, line feeds were included inappropriately and commas were placed at the beginning or end of an output, see Figure 18, compared with Figure 19. The problem was supposed to be caused by some "leftover" bytes in the device buffer or at the serial COM port. The Property Node determines the number of bytes so that they can be read (see above). The problem has not occurred again after implementing the Property Node coupled with the VISA Read function (Figure 19).

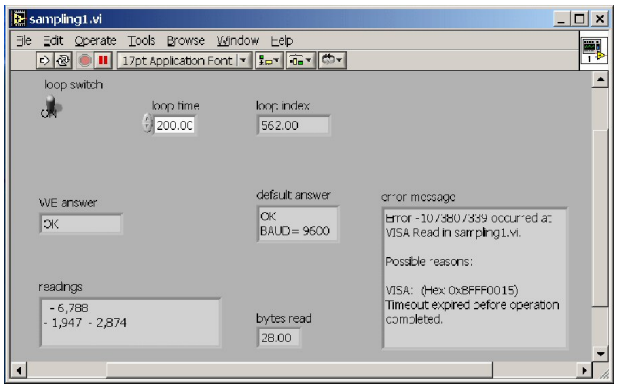


Figure 18. Switched LabVIEW^{TM} HMR2300 Driver Output.

The sequence structures were implemented because of a characteristic of LabVIEW^{TM}. LabVIEW^{TM} uses graphical programming. It offers functions in form of blocks. These blocks can be dropped into a block diagram and connected or "wired" together. This block diagram defines the data flow. Since most blocks have a variety of connectors that are not needed to be used in every application they can be left unconnected. But there exist also blocks that do not have to be wired to other executable blocks. One of these blocks is the Wait (ms) block. It has only one input that specifies the number of milliseconds to wait. In such cases it applies a different execution order than the data flow defines. In such cases of unconnected block connectors or stand alone blocks, they are mostly, but not necessarily, executed from left to right and top to bottom, based on their location within the block diagram [Ref. 19]. Wait (ms) blocks were placed in between connected blocks as a first attempt. Although the outputs were displayed correctly after starting the driver application the same problem as described above occurred (see loop indices in Figures 18 and 19). This was supposed to be caused by these two different block execution orders. The problem was fixed with the implementation of sequence structures. They define explicitly the execution order of blocks.

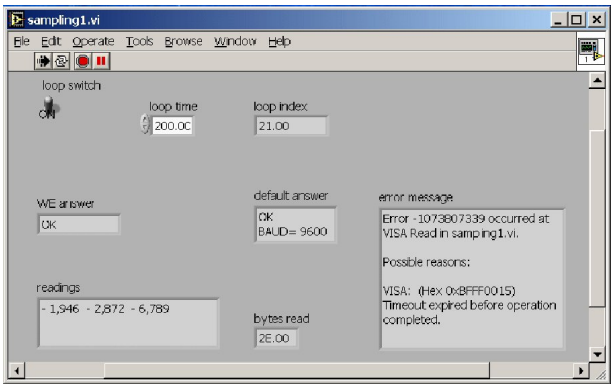


Figure 19. Proper LabVIEW^{TM} HMR2300 Driver Output.

In Figure 20 is shown only a principle sketch of the HMR2300 LabVIEW^{TM} driver structure because the used tools were implemented in three sequence structures and one while loop. The sequence structures hide most of the driver parts. The genuine block diagram would therefore need a lot of space if it would be "unfolded". But this principle structure sketch contains every information of the genuine block diagram. The LabVIEW^{TM} HMR2300 driver is part of the data CD of this report. For additional or more specific information about LabVIEW^{TM} characteristics, tools and functions refer to [Ref. 19].

3. LabVIEW^{TM} Driver Conclusions

Conclusions from implementing the HMR2300 LabVIEW driver are that the HMR2300 has some characteristics that have to be taken into account before it is used as a measurement device of embedded ACS (control algorithm) SIMULINK® Model hardware-in-the-loop simulations.

First of all, serial communication to the HMR2300 has to be terminated appropriately as long as it is not disconnected from its power source. In the case when measurements are interrupted by e.g. LabVIEW^{TM}'s Abort Execution button and restarted after a while the displayed output (in this case the output read from the

HMR2300 LabVIEW^{TM} Driver Structure

sequence structure

i = 0:	VISA Open	serial port	**Initialize**
i = 1:	Property Node	(number of bytes at port)	**Configure**
i = 2:	VISA Read	bytes from port	
i = 3:	VISA Write	write enable command	
i = 4:	wait (ms)	4	
i = 5:	VISA Read	write enable response	
i = 6:	VISA Write	default status command	
i = 7:	wait (ms)	4	
i = 8:	VISA Read	default status response	
i = 9:	VISA Write	write enable command	
i = 10:	wait (ms)	4	
i = 11:	VISA Read	write enable response	
i = 12:	VISA Write	output format command	
i = 13:	wait (ms)	2	
i = 14:	VISA Read	output format response	

while loop until manually switched of

sequence structure

i = 0:	VISA Write	output command	**Action**
i = 1:	wait (ms)	3	
i = 2:	VISA Read	measured **B** components	**Data**
i = 3:	wait (ms)	loop time delay	

sequence structure

i = 0:	VISA Close	serial port	**Close**
i = 1:	General Error Handler		

Figure 20. HMR2300 LabVIEW^{TM} Driver Principle Sketch.

device) might not be correct.

This applies also to each device response (see [Ref. 9]). Device responses have to be read after each command. If the device response `OK¬` after command `*ddWE<cr>` is not read, this response remains at the serial port or in the device buffer until it is read or the HMR2300 is switched off. Continuous output may be inconvenient because in this case all magnetometer readings have to be read at a device specific output rate, see Table XVI. The minimum sample rate of the HMR2300 is ten samples per second.

Output in BCD ASCII format may be inconvenient for data conversion from HMR2300 unit *counts* into SI unit Tesla, because this output format conforms to complex display rules.

Timing between send commands and read responses is important, as it is stated in [Ref. 9]. It can appear that commands are sent before the last response is read without included artificial time delays (Wait (ms)) or appropriate time-out settings. The same effect of wrong outputs as mentioned above occurs.

D. MATLAB SIMULINK® DRIVER

After the HMR2300 LabVIEW^{TM} magnetometer driver was implemented the next step has been to create and implement a magnetometer driver that was compatible with the ACS (control algorithm) SIMULINK® model. Additionally to Paragraph III.B.3 it has to be mentioned that only one xPC Target application can run on one target PC. The xPC Target applications can be built from any SIMULINK® model. All in- and output features have to be in this model to facilitate real data in- and output to and from the target application.

Both SIMULINK® and LabVIEW^{TM} use graphical programming. The difference is that SIMULINK® is a simulation tool and LabVIEW^{TM} is an application programming tool. LabVIEW^{TM} is intended to control real hardware devices, conversely SIMULINK® is not. However the MATLAB® product family provides some tools to communicate with serial devices. Unfortunately the used MATLAB® version was not licensed for the use of MATLAB®'s Instrument Control Toolbox.

So far, to establish correct serial communication between an xPC Target application and the HMR2300 magnetometer is not achieved yet. Some in-line testing was done to verify that commands are sent correct in HMR2300 requested format as well as to verify the correct responses from the device. As probes were set up at first, a LapTop with Windows® HyperTerminal and later on a PC running LINUX with MiniTerm that has provided more features than HyperTerminal. Either one or the other were linked in between target PC and and HMR2300 magnetometer and have displayed the current actual data that are passed between both devices. A principle sketch of the experimental setup can be seen in Figure 21.

As long as there are no encouraging results in developing an xPC/SIMULINK® HMR2300 driver it was neglected to spend any time on SIMULINK® implementations of data conversion from *counts* to Tesla. If

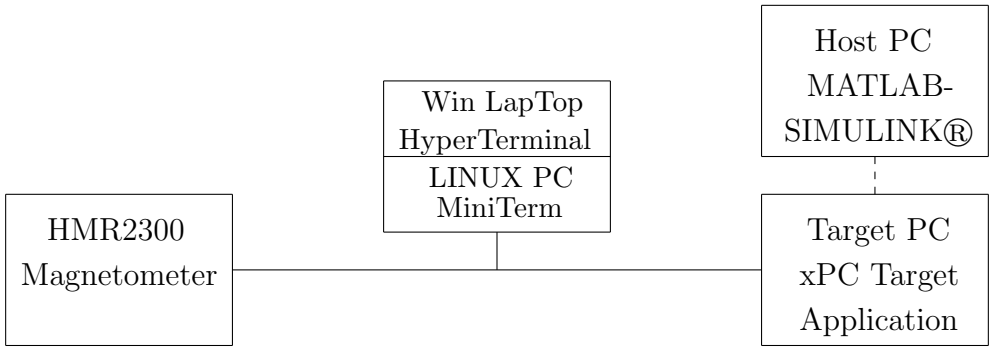


Figure 21. Experimental xPC Setup.

the HMR2300 magnetometer cannot be used with xPC Target applications this work would be obsolete.

1. xPC Serial Support

xPC Target offers a wide spectrum of facilities of data input and output, [Ref. 21], [Ref. 22]. The xPC Target I/O library includes also a set of SIMULINK® blocks for different approaches of RS232 communication. Blocks can be found for synchronous and asynchronous serial communication or for binary mode. If included in a MATLAB® installation, the SIMULINK® library browser provides access to section xPC.

a. RS232 Sync/Async Mode

The difference between synchronous and asynchronous mode is that the synchronous mode sends data or commands to a serial device and waits until the expected response is coming in. In the meantime this mode blocks or stops the execution of the xPC Target application until the whole response is read in or a time-out is reached.

The RS232 asynchronous mode does *not* block or stop the execution of the xPC Target application. It sends data or commands to the serial device. But the xPC Target application updates the output from the RS232 async block only when an entire package of data is received from the external device [Ref. 22].

The asynchronous mode was chosen for use. Because it is not wanted to block or stop the xPC Target application during the simulation waiting for data. The reasons for this are software-related. The real-time xPC Target application updates the output structures that contains data from the external device based on the chosen sample time of the xPC RS232 blocks and when data are received, see the short xPC RS232 block descriptions below, [Ref. 22]. All other blocks can be updated in the meantime.

The RS232 asynchronous mode blocks contained in section xPC Target, SIMULINK® library browser, are shown in Figure 22, [Ref. 22]. These blocks are necessary to fulfill the functions of an instrument driver, see Figure 16 and Paragraph III.B.1. These blocks can be treated like ordinary SIMULINK® Blocks. They can be directly dropped into a SIMULINK® model. The expectation was that these blocks that are created to act as parts of a RS232 driver would facilitate communication with the HMR232 magnetometer. Unfortunately they do not. This has different reasons. But first of all the use of these SIMULINK® supported xPC blocks is described briefly.

The actual xPC RS232 asynchronous SIMULINK® experimental model (Figure 31), its settings and parameters are contained in Appendix C.

RS232-Setup Block. This block is to setup a single RS232 COM port at the target PC. So each additional COM port would need another RS232-Setup block. This block send the initialize and termination messages [Ref. 22]. The RS232-Setup block covers both *Initialize* and *Close* function. The mask to define its block properties (that are in this case the properties of the serial port) is shown in Figure 32, Appendix C. Since

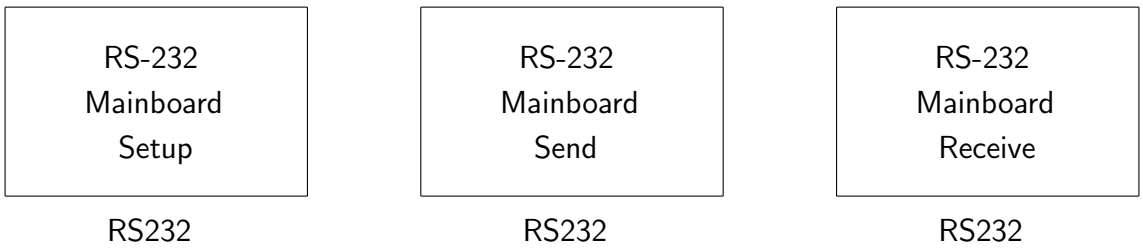


Figure 22. xPC RS-232 Driver Blocks (Asynchronous).

the RS232-Setup block does setup only the COM port and not the HMR2300 magnetometer it has to be set into an appropriate configuration before it is connected to the xPC Target application or with RS232-Send blocks within the target application. The second case means that the device would be configured each time the RS232-Send configure block is updated within the real-time execution of the xPC Target application, see below, RS232-Send block and [Ref. 22]. Except the SIMULINK® model of the xPC Target application is extended with an appropriate construct to sample just once at the beginning.

RS232-Send Block. This block is to send data or commands via the specified COM port. It covers *Action*, respectively *Configure*, functions. This block is sampled with a sample time within the xPC Target application [Ref. 20], [Ref. 21]. It was decided to sample this block once in a second for experimental purposes.

RS232-Receive Block. The RS232-Receive block receives data packages. The receive port does not have to be necessarily the same as the send port. This block is also sampled within one xPC Target application cycle. It is sampled once in a second to fit the sampling of the RS232-Send block. Both RS232-Send and RS232-Receive block provide a time-out function. The order of execution is mostly defined by the RS232-Send-Receive Message Structures.

RS232 Async Message Structures. These RS232-Send and RS232-Receive blocks

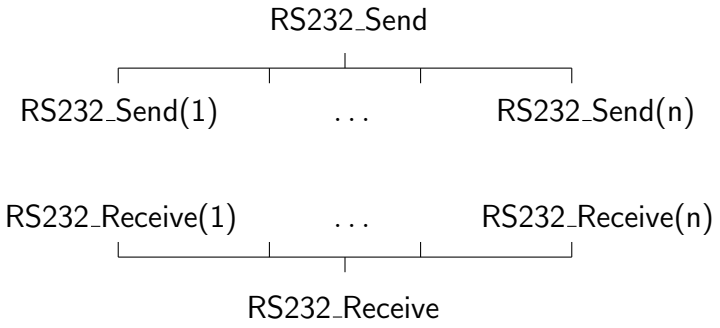


Figure 23. RS232 Message Principle.

have to be initialized. That means that send and receive structures have to be written. These structures specify the commands that are sent and their expected corresponding responses. They are created in form of an M-file. By loading this M-file into the MATLAB® work space and running the SIMULINK® model, both the RS232-Send and RS232-Receive block are updated with these structures. After compiling the SIMULINK® model with MATLAB Real-Time Workshop® and downloading to the target PC the xPC Target application is ready for executing these structures and to communicate with an external serial device. The principle of these message structures is shown in Figure 23, [Ref. 22]. The M-file implementation of such message structures is shown in Table XXII, Appendix C. The shown example is one of few message structures that have received at least an incomplete response from the HMR2300.

xPC/SIMULINK® Async Experimental Driver Structure.

The RS232-Receive block does not block/stop the xPC Target application execution while waiting for the device response that corresponds to a previous send command. So there is no guaranty that each received response corresponds to the command sent just before, although send and receive ports are the same. If there occurs a time delay of a response, caused by whatever, the order of interpretation of

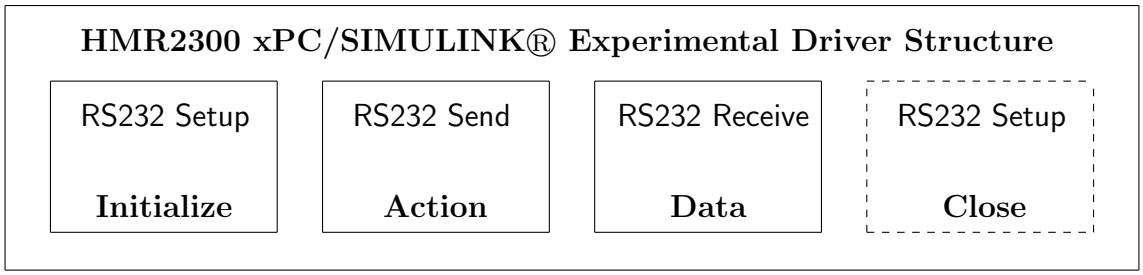


Figure 24. xPC/SIMULINK® Experimental Driver Structure.

these responses has to fit or the same problem as described in Paragraph III.C.2 occurs. To avoid this only one send message is implemented. It causes output (`*00P\r`, respectively `*00#\r` for experimental purposes) and configures the HMR2300 before it is connected to the target application. The HMR2300 magnetometer feature to store user defined settings in its EEPROM has been very helpful in this case [Ref. 9], [Ref. 10]. Another possibility would be to implement different RS232 Send blocks executed in a definite order with appropriate time delays. To keep things simple this was not done. So the experimental SIMULINK® driver model contains just one send message structure and just one receive message structure. After setting appropriate time-outs and a sample time of one second there have been no timing problems. The configuration of the HMR2300 was done with the LabVIEW^{TM} driver and confirmed with the HMR2300 demo software. The xPC/SIMULINK® Experimental Driver Structure can be seen in Figure 24.

xPC RS232 Async Mode Experiences.

Usually corresponds to each sent command an answer. There is not necessarily an answer because the RS232-Receive block is independent from the RS232-Send block. Unfortunately the documentation to RS232 message structures is not that useful and technical support (via e-mail and telephone) from Mathworks has resulted often in unsatisfying and unclear statements. E.g. the *End of Messages* (EOM)

indicator in RS232 synchronous message structures was specified in MATLAB® documentations [Ref. 22] as it is explained in Table XXII, Appendix C. Mathworks technical support has it specified as actual end-of-message indicating character, not as a number. Typing `'\r'` instead of `1` has lead to an xPC Target error message "`ERROR: RS232 Send/Rec: receive data error`".

The RS232 asynchronous send/receive structure shown in Table XXII, Appendix C, is to send the `*00#\r` command that should result in the serial number of the HMR2300 device. It conforms to the command set shown in Table XVI, Appendix A. This response is independent from the chosen output format (BCD ASCII/BINARY). The correct response of the used HMR2300 magnetometer on this command is "`SER# 1263 6827`".

The problem that has occurred and is still unsolved are the supported data types for RS232 synchronous message fields, [Ref. 22]. They are listed with their documentations explanation in Table IX. As one can see there is no "%s" that would facilitate the response format BCD ASCII string. Since LabVIEW^{TM} it's VISA Read tool has expected the number of bytes to be read, it was not necessary to specify the response as string. The xPC RS232-Receive block is coupled with such message structures as shown in Table XXII, Appendix C. These structures facilitate only the use of data formats as shown in Table IX. At this point started a long lasting process of testing different combinations to read in at least one sample of measurements, no matter in what format was. During this process this work was supported from Jim Horning, software engineer at the SSAG.

At first the send message structure was verified. The Laptop with WINDOWS® HyperTerminal and later on the LINUX PC with MiniTerm have displayed the correct send commands `*00#\r` respectively `*00P\r`. After that the correct answer of the HMR2300 device was verified in the same way as the command. Both different computers linked between HMR2300 and target PC have displayed

Format	*Description*
%c and %C	single character and wide character
%d or %I	signed decimal integer
%u	unsigned decimal integer
%o	unsigned octal integer
%x or %X	unsigned hexadecimal integer, using '`abcdef`' or '`ABCDEF`' for hexadecimal digits
%e or %E	exponential format using
%f	floating point
%g	signed value printed in `f` or `e` format depending on which is smaller
%G	signed value printed in `f` or `F` format depending on which is smaller

Table IX. Supported Data Types for xPC RS232 Messages.

the *correct* response "`SER# 1263 6827`", respectively one set of $\{\xi,\ \eta,\ \zeta\}$ **B** vector component readings. So the actual problem is the incompatibility of HMR2300 sent response format and MATLAB® xPC/SIMULINK® supported response format. A couple of attempts were made to find a fitting combination to read in the correct `*00#\r` response. A few of them are listed in Table X.

One can see that the first three receive commands grab the first four digits of the HMR2300 serial number. Interesting in Table X is the fourth attempt that lead to the result `6827`, actually the second four digits of the serial number. This has resulted in some experiments with the command `sscanf` in C. If the `sscanf` (read from string) function is implemented in the same way as it is in C then both commands `sscanf(`*source*`,%u%u)` and `sscanf(`*source*`,%u %u)` should result in two unsigned integers received from a string. Actually this implemented function does not. The question to Mathworks concerning known or possible bugs with the implementation of this function was answered negatively.

The attempt to read in Binary coded magnetometer readings with the

	Response Format	*Attempt*	*Result*
1	ASCII	`RS232_Receive(1).RecData = 'SER\# %u\r';`	1263
2	ASCII	`RS232_Receive(1).RecData = 'SER\# %u \r';`	1263
3	ASCII	`RS232_Receive(1).RecData = 'SER\# %u%u \r';`	1263
4	ASCII	`RS232_Receive(1).RecData = 'SER\# %u%u\r';`	6827
5	Binary	`RS232_Receive(1).RecData = '%X%X%X%X%X%X\r';`	-

Table X. Message Structure Example.

fifth example in Table X was as unsuccessful as the other shown attempts. The resulting MATLAB® displayed output is not shown here, since it was not expected to get reasonable values with applying a text based function to binary format. However it was tried. The RS232 Receive block was connected with one SIMULINK® Outport block, see Figure 31, Appendix C. That block creates an array that contains simulation time steps and received data and logs this array into MATLAB®'s work space, [Ref. 20], [Ref. 21]. These arrays had the wrong dimension ($n \times 1$ instead of $n \times 6$ for six time steps) and were filled with random numbers (displayed format was Hex). Another attempt to read in 28 BCD ASCII coded bytes interpreted as a line of characters and one carriage return (similar to the fifth attempt for Binary format in Table X) crashed the target PC.

xPC RS232 Asynchronous Mode Conclusions

The conclusion from these time-consuming "trial & error" experiments with SIMULINK® supported serial asynchronous xPC RS232 Send and RS232 Receive blocks is that correct serial communication in this configuration is not possible at the moment. There may be a possible solution to combine supported data formats and get a correct result. But it is not found yet. A further supposition is that the implemented `sscanf` function is not working correctly.

b. RS232 Binary Mode / RS232 Hybrid (Async/Bin)

There exist another set of SIMULINK® supported blocks for xPC serial communication. During experiments with xPC asynchronous serial blocks was switched shortly to the xPC binary serial blocks. They can be found also within the SIMULINK® library browser, section xPC, subsection RS232. The settings of these blocks are shown in Appendix C.

Since it is not wanted to send the HMR2300 commands in binary mode it was considered only the RS232 Setup, the RS232 Binary Receive and the Unpack block. The RS232 Setup block is exactly the same as shown in the last Paragraph III.D.1.*a*.

RS232 Binary Receive Block. The RS232 Binary Receive Block reads in packages of Binary coded data. This block needs a specified maximum length of the data package that is to receive. Its output is always a vector with a width corresponding to this maximum package length. Furthermore this block needs an input at the Enable connector to switch this block on and can output at its Done connector a function call. Because of the requested number of bytes to be read was expected that this block would act similar to the LabVIEW^{TM} VISA Read function, see [Ref. 19], Paragraph III.C.2.

Unpack Block. The Unpack block belongs to the family of xPC UDP/IP blocks. UDP is the *User Datagram Protocol*, see for further information [Ref. 22]. The Unpack block brakes a received data packet into a specified format.

xPC RS232 Hybrid Experiences

The SIMULINK® implementation of this model is also contained in Appendix C. The experimental SIMULINK® driver model with asynchronous RS232

Time	*1st Column*	*2nd Column*	*3rd Column*
1	0000000000000000	0000000000000000	0000000000000000
2	0000cdefecf312f7	0000000000000000	0000000000000000
3	0000efecf312f70d	0000000000000000	0000000000000000
4	0000ecf312f70dcd	0000000000000000	0000000000000000
5	0000f312f70dcdef	0000000000000000	0000000000000000
6	000012f70dcdefec	0000000000000000	0000000000000000
	4th Column	*5th Column*	*6th Column*
	0000000000000000	000000001003bb90	0000000000000006
	0000000000000000	000000001003bb90	ecf312f700000006
	0000000000000000	000000001003bb90	f312f70d00000006
	0000000000000000	000000001003bb90	12f70dcd00000006
	0000000000000000	000000001003bb90	f70dcdef00000006
	0000000000000000	000000001003bb90	0dcdefec00000006

Table XI. Binary Read Result Sample.

Send block, RS232 Binary Receive block and Unpack block (Figure 35, Appendix C) has lead to results that have looked promising at the first. The read data logged to the MATLAB® workspace seemed to correspond somehow to expectations. The $(n \times 6)$ array was fully filled and in each column were contained repeating values. Later on these numbers turned out to conform to no known rules. And from that moment on as result arrays appeared as the one shown in Table XI, there was no longer a thought that there is any relationship to the data sent by the HMR2300 magnetometer. This array shows both in first and sixth column cyclic changing hex numbers. Conversely the fifth column contains constant values. All other columns contain only zero. However Mathworks Technical Support was asked if there is any data interpretation or formatting by MATLAB® and its packages of received data before they are displayed or logged. The answer was negative.

The conclusion from experiments with this hybrid xPC/SIMULINK® RS232 async/bin model is, therefore, that the command sets are sent to the magnetometer correctly. The inline testing PC linked between target PC and magnetometer

has confirmed it. The response of the HMR2300 is as expected and stated in [Ref. 9]. Because of such random displayed "response" structures it is unclear whether xPC Target reads in data from the device or not. Further it is unclear whether there is any interpretation or formatting by xPC/SIMULINK® after data are received and before they are passed for further analysis.

2. MATLAB® Serial Support

An interesting fact is that although SIMULINK® supported xPC Target RS232 blocks facilitate correct communication only in one direction "send" from target PC to HMR2300 magnetometer, MATLAB® facilitates correct communication in both directions. In [Ref. 23] basics of serial communication and how to use MATLAB® to communicate to serial devices are described. Therefore exist a couple of commands, see [Ref. 24]. In Table XII is shown briefly how MATLAB® supports serial communication. This example is self explanatory.

Matlab® Serial Communication Experiances

Some testing was done with serial MATLAB® communication. E.g. the continuous output command `*00C CR` was sent and it was tried to sample the data stream manually with `fscanf(s)`. The effect was that a moved HMR2300 has not resulted in an appreciable change of the displayed readings. This is uncommon considering to the device accuracy.

These few experiments have confirmed conclusions obtained with the LabVIEW^{TM} driver. The HMR2300 response has to be read after each command. Otherwise this response remains in the device buffer/ at the serial port and blocks all other responses. That means for continuous output (`*00C CR`) that the whole data stream has to be read from the device at the sample rate. These data have to be stored somehow e.g. in just one sample that is overwritten each time the next sample

is read in.... This would be compatible also with an easy implementation of the driver structure e.g. such as in the xPC/Simulink® async or hybrid experimental models, if they were working properly.

3. SIMULINK S-Functions

Since MATLAB® facilitates correct serial communication with the HMR2300 magnetometer but SIMULINK® supported xPC blocks do not, the next attempt was to customize SIMULINK® blocks with MATLAB s-functions.

The SIMULINK® library browser contains in section SIMULINK, subsection User Defined Functions different blocks for customizing. One of these blocks is the S-Function block. S-functions can be implemented as M-files.

A citation from [Ref. 25] follows: "An *S-function* is a computer language description of a Simulink block. S-functions can be written in MATLAB®, C, C++, Ada, or Fortran." ..."S-functions allow you to add your own blocks to Simulink models. You can create your blocks in MATLAB®, C, C++, Fortran, or Ada. By following a set of simple rules, you can implement your algorithms in an S-function." ..."The most common use of S-functions is to create custom Simulink blocks. You can use S-functions for a variety of applications, including:

- Adding new general purpose blocks to Simulink
- Adding blocks that represent hardware device drivers
- ...".

This sounds very promising. But the attempt of writing M-file S-functions for serial communication was not successful. All information concerning M-file S-functions that were available at MATLAB®'s documentations [Ref. 24] and SIMULINK®'s documentations [Ref. 25] have dealt with implementing of mathematical functions. And it seemed that the "set of simple rules" mentioned above fits

MATLAB® Command *MATLAB® Output*

```
s=serial('COM1','Terminator',
        'CR','BaudRate',9600)
                              Serial Port Object : Serial-COM1

                              Communication Settings
                                  Port:               COM1
                                  BaudRate:           9600
                                  Terminator:         'CR'

                              Communication State
                                  Status:             closed
                                  RecordStatus:       off

                              Read/Write State
                                  TransferStatus:     idle
                                  BytesAvailable:     0
                                  ValuesReceived:     0
                                  ValuesSent:         0

fopen(s)
frintf(s,'*00P')
[read,count]=fscanf(s)
                              read=

                               3,393  -   821  - 4,096

                              count =

                                 28
fclose(s)
```

Table XII. MATLAB® Serial Communication Example.

only this purpose. A demo S-function or other examples that correspond to this task are not yet found.

However it was tried to translate these steps of "Writing M S-Functions" ([Ref. 25]) and use MATLAB® commands for serial communication ([Ref. 24], also Paragraph III.D.2). The results have not been encouraging. Even if there have not appeared error messages after loading the M-file into MATLAB®'s work space they have appeared during updating the SIMULINK® S-function model.

But work on this topic is not finished. Another possibility is to have a closer look at S-functions written in different languages, e.g. C S-functions, not constraint by MATLAB® structures. Further a look at the S-function implementation of the xPC RS232 blocks is self-evident. Maybe there can be found a way to customize xPC/SIMULINK® RS232 with this approach.

4. xPC/SIMULINK® Driver Conclusions

Even direct progress to solve the task to communicate to the HMR2300 magnetometer with xPC/SIMULINK® blocks and customized S-functions blocks was not obtained yet, some information were obtained about an useful implementation depending on xPC/SIMULINK characteristics. Further some not working approaches were eliminated and also was decreased the number of remaining possible approaches to solve the task of using the serial HMR2300 magnetometer with the xPC Target application of the ACS control algorithm SIMULINK® model.

The use of genuine xPC/SIMULINK® RS232 blocks does not facilitate accurate and faultless communication. MATLAB® facilitates this but there was no success yet to implement MATLAB® command structures into MATLAB® S-functions and customize SIMULINK® blocks. Remaining possibilities of this approach are:

- customizing SIMULINK®'s S-Function blocks with S-functions of other programming languages than that one used in MATLAB® M-files, e.g. C, C++, Ada, or Fortran,

- customizing xPC RS232 block implemented S-functions to fit HMR2300 response format depending necessities,

There may be more possibilities within MATLAB®'s Instrument Control Toolbox. If Mathworks offer a trial license of that toolbox one may check if it is useful. But to buy a new license without any guaranty of success is senseless.

Another possible solution is to step back from the use of the HMR2300 magnetometer and switch back to the Schonstedt Instrument CO. SAM-73C magnetometer. This means the use of additional analog/digital conversion hardware.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. CONSIDERATIONS ABOUT THE AIR-BEARING TABLE SETUP

In this chapter is given only a brief overview of the hardware-in-the-loop simulation hardware setup. This hardware test setup is a large project within the process of building NPSAT1. Although NPSAT1 is the second spacecraft built by the SSAG it is quite different from PANSAT. The ACS SIMULINK® Model hardware-in-the-loop simulation is one of the new projects. Therefore, the hardware test setup has been and still is a challenge since there has been no similar project in the SSAG. Nearly all engineers of the SSAG are involved. So the hardware setup is characterized with teamwork and interdependence.

The process of setting up the SSAG air-bearing table was a process of experiencing a wide spectrum of different concerns, handling and solving problems in teamwork and trying to keep on schedule. Keeping on schedule was not successful in all cases. There are different reasons for this and most of them are just part of the reality of systems engineering. So the opportunity was offered to experience a couple of different approaches to deal with the reality of systems engineering.

Furthermore, necessary information are stated in this chapter to assemble, setup and use the air-bearing platform in hardware-in-the-loop simulations.

A. OVERVIEW OF THE HARDWARE-IN-THE-LOOP SIMULATION SETUP

A short description of the intention of these hardware-in-the-loop simulations is given in Section I.C. Now are described different components of the hardware-in-the-loop simulation test setup. A principle sketch can be seen in Figure 25. The major components and the most important data are shown. They can be necessary

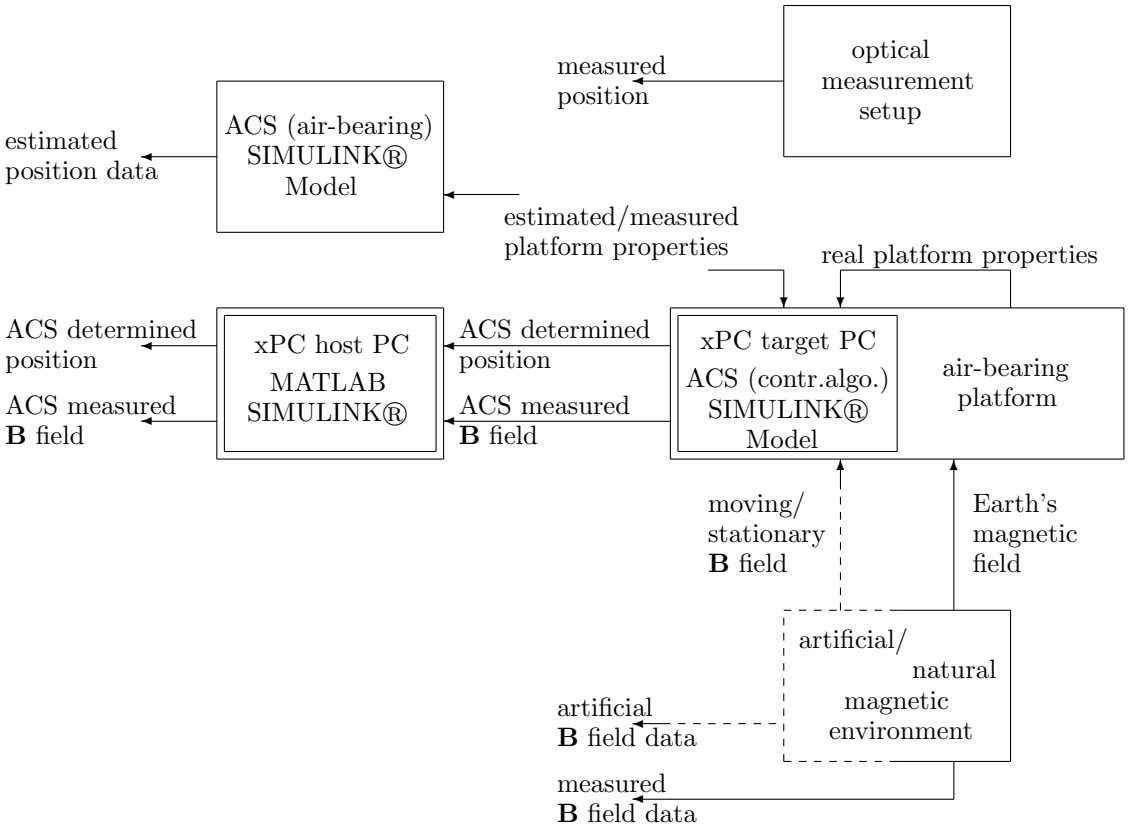


Figure 25. Hardware-in-the-Loop Simulations Setup Principle.

whether for air-bearing platform position analysis and interpretation or for error search/handling. A short description of these major components follows.

Air-bearing Platform. The air-bearing platform is the actual hardware model that is to be controlled by the xPC Target ACS (control algorithm) SIMULINK® Model application that runs in real-time. Components of the air-bearing platform are:

- platform structure,
- xPC target PC (embedded ACS (control algorithm) SIMULINK® model, data connections),

- electric power supply (electrical power sources as well as power distribution setup),
- magnetic setup (magnetic torque rods, torque rod driver, magnetometer),
- components of the laser measurement setup.

Information about the platform structure can be found in [Ref. 8] and in Sections IV.B, IV.C of this script. Information about the electric power supply, the magnetic setup and components of the laser measurement setup are available at the SSAG (see contact information in [Ref. 1]).

The measurable actual dynamic behavior of the air-bearing platform is influenced by its real mass properties, namely the location of the center of mass with respect to the center of rotation or the properties of the tensor of inertia. Since it is controlled with magnetic torque rods, the difference between real magnetic environmental data and measured magnetic environmental data also influence the "attitude" of the air-bearing.

xPC Target PC/ACS (Control Algorithm) SIMULINK® Model. To differentiate between used ACS SIMULINK® models they were labeled each with its implemented system. The ACS (control algorithm) SIMULINK® Model is the embedded part of the ACS SIMULINK® Model. The principles of the embedded ACS (control algorithm) SIMULINK® Model are described in [Ref. 7]. Only a basic overview is given, since hardware components have changed and may change again (see Chapter III).

This model contains the SIMULINK® implementation of the magnetic control algorithm that is expected to be verified, see also Chapter I. Furthermore all necessary additional SIMULINK® implementations to deal with real input (magnetic sensors) and real output (magnetic torque rods) are included.

This model is compiled with MATLAB Real-Time Workshop® into a SIMULINK® xPC Target real-time application and downloaded (via wireless Ether-

net connection) to the xPC Target PC. The target PC is the hardware that runs the ACS (control algorithm) SIMULINK® Model and is connected with real magnetic sensors and magnetic actuators. For information about using SIMULINK® xPC Target refer to [Ref. 20], [Ref. 21].

This "embedded" system is fed with *estimated* or *measured* air-bearing platform properties and has to deal with *real* air-bearing platform properties. Furthermore its magnetic sensors measure data from an artificial magnetic **B** field (intended for later tests) or the Earth's magnetic field. The better the estimated or measured air-bearing properties and the measured magnetic field data matches the real ones the easier and clearer will be the comparison of estimated dynamical platform behavior (ACS (air-bearing) SIMULINK® model) and measured dynamical platform behavior.

Optical Measurement Setup. The optical measurement setup consists of:

- Laser pointers and indicator screen,
- CCD camera,
- PC with analysis software.

Information about the optical measurement setup are available also at the SSAG (see contact information in [Ref. 1]). The laser pointers are mounted on the air-bearing platform and generate a pattern at the indicator screen. This pattern moves with respect to the motion of the air-bearing platform. It is monitored and recorded on the transparent indicator screen with the CCD camera. These data of laser point patterns are converted into position data of the air-bearing platform.

The accuracy of these measured position data depends on the measurement conditions and the analysis algorithm. One question is at the moment the synchronization of those measured position data with xPC Target PC's logged, ACS (control algorithm) SIMULINK® determined, position data. xPC Target stores the position

data (determined from the ACS (control algorithm) SIMULINK® model) and downloads them after finishing the execution of the real-time application. They can be logged together with their corresponding xPC Target time. The measured position data can be logged with their corresponding system time. For analysis is important to know the relationship between those two system times or to initialize them simultaneously.

xPC Host PC. The xPC host PC is the computer on which the MATLAB SIMULINK® environment for xPC Target is running. The ACS (control algorithm) SIMULINK® model is compiled with Real-Time Workshop® on this computer. This is also the computer where all xPC target PC data are downloaded to, if they are specified for logging, see [Ref. 20], [Ref. 21].

ACS (air-bearing) SIMULINK® Model. The ACS (air bearing) SIMULINK® Model is the reference for the hardware-in-the-loop simulations. It is the SIMULINK® simulation of ACS control algorithm, magnetic sensors, actuators and environment and air-bearing table dynamics. It is also the source of the embedded ACS (control algorithm) SIMULINK® model. The comparison of simulation results (*estimated position data*) obtained from the ACS (air-bearing) SIMULINK® model with *measured hardware-in-the-loop simulation results* obtained from the optical measurement setup and data obtained from the embedded system verifies the correct SIMULINK® implementation of the ACS control algorithm.

The ACS (air-bearing) SIMULINK® model is fed with *estimated/measured* platform data. So the better the used platform properties match the real platform properties, the better this simulation will match the actual test results. Information about the ACS SIMULINK® Model can be found in [Ref. 5], [Ref. 7].

Magnetic Environment. It is planned to use an artificial magnetic environment for advanced tests. This artificial magnetic field could simulate the magnetic field changes that NPSAT1 will experience in orbit. Hardware-in-the-loop simulations will be done

with the Earth's magnetic field at present. The process of measuring and analyzing the Earth's magnetic field in the SSAG laboratory is described in Chapter II.

It could be convenient for analysis or error search/handling to compare the known magnetic field data with magnetic field data measured and used by the embedded ACS (control algorithm) SIMULINK® model. This is necessary to get to know whether the magnetometer is sampled within the magnetic decay of the previously pulsed torque rods or if there is any appreciable other influence on magnetometer readings.

B. AIR-BEARING PLATFORM MASS PROPERTIES

Since not all components are fully finished only some estimated properties can be presented. However these estimated values or rather the corresponding mathematical model can be used during the process of adjusting the air-bearing mass properties.

1. Considerations about Real Mass Properties

The process and approach of designing the air-bearing platform is described in [Ref. 8]. The design of the air-bearing table was done trying to fit requirements based on the equations of motion of NPSAT1. The relationship of moments of inertia of both NPSAT1 and air-bearing platform should be comparable, see Equation I.1. The tensor of inertia **I** should be a diagonal tensor:

$$\mathbf{I} = \mathbf{diag}(\mathrm{I}_{ii}),\ i \in \{\xi,\ \eta,\ \zeta\}. \tag{IV.1}$$

The approach to meet the requirements was to design the platform strictly symmetrical. This should result in vanishing products of inertia and avoid a lot of counterweights additional to all necessary components.

Since there were some changes in components and component housings, this strict symmetrical design could not be achieved in all cases. The attempt is now to balance it out without additional counterweights because of constraints on space and to keep as much of the properties of the symmetrical basic structure. The first check on the basic structure was promising. The air-bearing platform was placed on its pedestal without the magnetometer, wireless Ethernet bridge, target PC, torque rod driver board, electric power supply electronics, their housings and without wire-harness. The deviation of its ζ axis from the perpendicular axis was less than $10 degrees$ just after bolting it together and without paying attention to aligned components. After shifting two batteries some millimeters in their boxes there was no appreciable deviation visible any longer. This may be a first sign that the approach has worked out. But it gives also a first idea how fragile the air-bearing mass properties are. And it has stimulated also a discussion on how to measure the air-bearing mass properties (real location of the center of mass, real moments and products of inertia).

As one can see in Figure 25 and as mentioned in Section IV.A there are a couple of reasons for measuring the real air-bearing mass properties. The ACS (air-bearing) SIMULINK® model simulates the dynamical behavior of the air-bearing platform based on parameters like mass properties. If these parameters do not fit (with some tolerance) to real values, the simulation results serve only academic purposes. These simulation results could not reasonably be used as reference for hardware-in-the-loop simulations.

The ACS (control algorithm) SIMULINK® model is used as in hardware-embedded system. It determines the necessary output to real actuators based on real input from sensors, conversely to the ACS (air-bearing) SMULINK® that uses data files and mathematical models of actuators. But both are functioning based on the implemented dynamics of the air-bearing. The problem is that the ACS (control algorithm) SIMULINK® model controls not a mathematical model of the air-bearing but the air-bearing platform with its real mass properties. The controller would not

be useful for the controlled system.

2. Estimated Air-bearing Properties

Quite useful air-bearing platform properties can be presented, obtained with the final mathematical model of the air-bearing platform. However these values should not be used as reference for actual hardware-in-the-loop simulations. The mathematical model can be used to estimate values and ease the process of adjusting the air-bearing platform mass properties.

The mathematical model is again a Maple 7.0 work sheet and explained in Appendix E. The values stated here are determined with "initial positions". That means coordinates were taken which were used to define the dimensions of the air-bearing platform. After the corresponding outer geometry was defined and the mathematical model showed that the mass properties would meet the requirements, the geometry was extended. To these initial positions was added some space to achieve adjustability. So these initial positions are not directly visible on the air-bearing platform.

But they give a coarse impression of the mass properties of the air-bearing platform. Furthermore they serve this purpose and there is only little use in considering estimates of different configurations of the air-bearing platform. If one wants to know the estimated values for particular component positions one may use the Maple 7.0 work sheet in combination with air-bearing platform technical drawings.

The in Table XIII shown values represent nearly the actual appearance of the air-bearing platform. They are determined with those values that are contained in the Maple 7.0 worksheet.

C. AIR-BEARING PLATFORM ASSEMBLY COMMENTS

All necessary information to assemble the air-bearing platform can be found in air-bearing platform assembly drawings, components list, single part drawings that

Estimated Mass Property	*Value*
mass m_{result}	$56.780\ kg$
ξ_{cm}	$-.469e-3\ m$
η_{cm}	$-.306e-3\ m$
ζ_{cm}	$.771e-2\ m$
$I_{\eta\eta}$	$2.7917\ kgm^2$
$I_{\xi\xi}$	$2.6746\ kgm^2$
$I_{\zeta\zeta}$	$2.1065\ kgm^2$

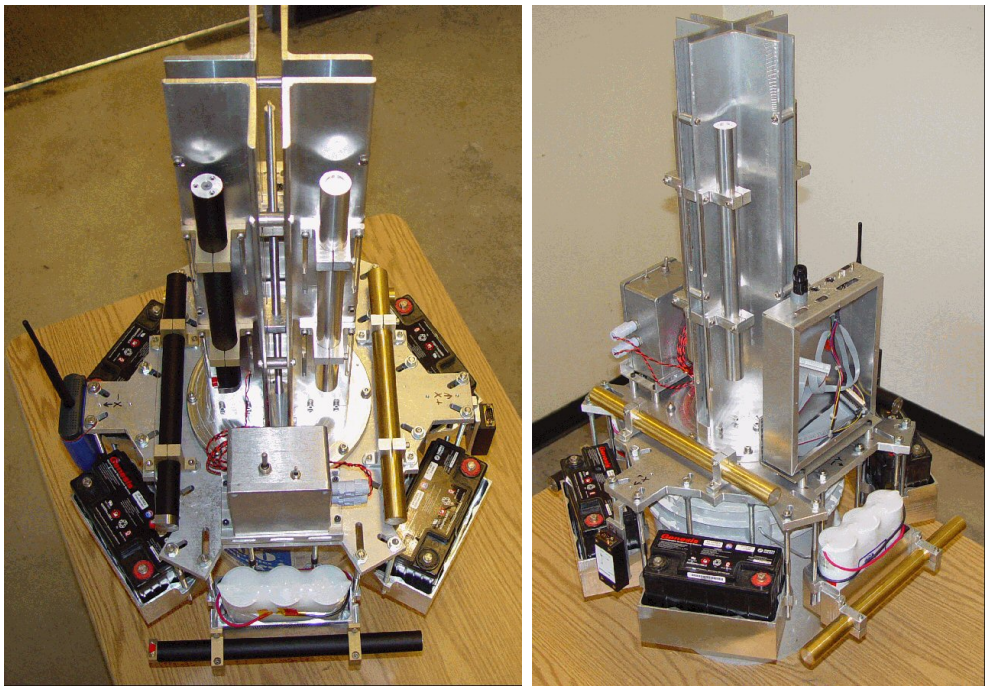
Table XIII. Estimated Air-bearing Mass Properties.

are contained on the data CD. A list of all technical drawings/datasheets concerning the air-bearing platform is shown in Appendix D.

However two pictures can be presented that show the basic intent of component arrangements on the air-bearing platform. Furthermore, the assembly process is described very briefly, for further information refer to the technical drawings. A series of photos were taken to document the assembly because no drawing can present this information in a way that is that convenient. Some of these photographs are also part of the data CD.

General. One has to be cautious every time the air-bearing inner part (convex) is handled. Scratches, dings, dents etc. may affect the functionality of the air-bearing table. The first assembly was done on a plastic bucket and this has worked out. The plastic bucket can be seen in the right part of Figure 26.

Basic Plate. The hexagonal part of the basic plate can be placed concentrically on a plastic bucket or something similar. This offers access to all necessary locations on the hexagonal plate. The air-bearing inner part has to be mounted to the disk. The disk can be seen in the left part of Figure 26, between two rods mounted to the hexagonal plate. The next step is to mount the disk at the center of the hexagonal plate. This completes the basic plate.



left: arrangement of magnetic torque rods (black), electric power supply box, wireless Ethernet bridge

right: arrangement of dummy rods, cpu box, HMR2300 magnetometer

Figure 26. Air-bearing Platform Assembly.

Battery Boxes. The 5/16 - 24 threaded rod pieces have to be mounted to the tapped holes in all battery boxes. They do not need to be secured. The excess length depends on what location of the center of mass/moments of inertia one wants to reach, see Appendix E. The order of mounting is arbitrary with one exception. The larger EP battery boxes are mounted on both sides of the two "teeth" of the hexagonal plate that have symmetrical mounting slots, see the left part of Figure 26.

Angle Uprights. The upright angles are mounted together with the structural parts of the angle cross. Both adjustable cross sections have to be aligned with the long slots of the angle uprights. The "T" bottom part can be mounted to the basic plate aligned with the axis that is given by both teeth with symmetrical slots. The angle

cross structure has to be mounted to that "T" bottom part. Those angle legs with long slots have to be aligned along the axis given by both teeth with symmetrical slots.

Magnetic Torque Rods. They are located perpendicular and next to each other on one side of the air-bearing platform. They have to be aligned with the air-bearing principle axes, Figure 38, [Ref. 27]. The location of magnetic torque rods can be seen in the left part in Figure 26.

Dummy Rods. These are the only counterweights at this time. They balance the magnetic torque rods. The batteries are balanced by their symmetrical arrangement. Electric power supply box and CPU box do not have the same dimensions but roughly the same weight when they are finished.

There are two different parts of dummy rods. Two dummy rods have a brass outside (golden color), three dummy rods have an aluminum outside (silver color). Both brass outside dummy rods balance the horizontal aligned magnetic torque rods. The other three dummy rods balance the magnetic torquer that is mounted on to the angle uprights, see Figure 26.

Magnetometer/Wireless Ethernet Bridge. The magnetometer (whether the HMR2300 or its replacement) is mounted at one "tooth" of the hexagonal plate that has symmetric slots and is located in the opposite direction of the magnetic torque rods.

The wireless Ethernet bridge is mounted at the opposite hexagonal plate "tooth" to the location of the magnetometer. Magnetometer and wireless Ethernet bridge do not have the same weight, dimensions and are not mounted symmetrically. Their products of inertia and their influence on the location of the center of mass can be neglected, see the current estimates in Table XIII.

EPS and CPU Box. Both are mounted above the smaller J-Cell battery boxes each on a custom mounting structure.

D. AIR-BEARING SETUP RECOMMENDATIONS

Some recommendations concerning the air-bearing table setup are stated in this section. Some of them were already mentioned but given again to concentrate the information.

Magnetometer. The magnetometer (no matter what particular device is used) should be located away from electro-magnetic noise caused by switching circuits etc. Thats why those magnetic torque rods are kept together on one side of the platform. It may be useful to measure magnetic field data with a magnetometer that is mounted on the air-bearing platform at rest with fully powered and functioning components and compare them with previously measured magnetic field data. This should yield information on whether the magnetometer readings are influenced / disturbed (e.g. by sampling within the magnetic decay of the pulsed torque rods).

Magnetometer axes have to be aligned with the air-bearing coordinate system (see Figure 38, Appendix E). If this is not possible, a principle axes coordinate transformation has to be done.

If magnetometer axes labels do not match air-bearing axes labels, a coordinate transformation has to be implemented. This can be as easy as switching output connections or additional mathematics within the embedded ACS (control algorithm) SIMULINK® model, e.g. Equation II.3.

Magnetic Torque Rods. The components of the applied magnetic dipole moment **m** are produced with magnetic torque rods. The embedded ACS (control algorithm) SIMULINK® model determines these components aligned with platform principle axes. Each of these magnetic torquers has to be aligned with an air-bearing platform principle axis. The direction sense of a particular torquer can be seen in the Design Description and Operating Manual, [Ref. 27].

The torque rod pulse cycle has to match the magnetometer sample cycle to avoid contaminated measurements. This can be ensured with sampling the mag-

netometer appropriately with respect to the magnetic decay time constant and the chosen time delay.

Air-bearing Platform Mass Properties. There is a high grade of interdependence between component locations, location of center of mass and inertia properties. As the air-bearing platform is short on space (reasons for that are given in [Ref. 8]), there is no space to add more components without affecting seriously the air-bearing mass properties. The use of whether the Maple 7.0 script in Appendix E or a different approach may be useful to check the estimated mass properties after changes in component locations.

Further a measurement of the real mass properties of the air-bearing platform is necessary. Firstly to check the quality of values estimated with the Maple 7.0 script in Appendix E. Secondly to use them in hardware-in-the-loop simulations because they influence the quality of the reference simulation results (ACS (air-bearing) SIMULINK® model) as well as measured results (air bearing platform/embedded ACS (control algorithm SIMULINK® model)).

The air-bearing platform was designed to be symmetrical to make all products of inertia vanish. This means symmetry in air-bearing mass distribution as far as possible and reasonable (location of components, mass density, etc.). It could be convenient because of this to use more than one of the various facilities to adjust components and change their positions symmetrically.

The intention was to fit the air-bearing platform to ACS SIMULINK® model hardware-in-the-loop simulations. However its modular design facilitates various applications.

Data Synchronization. It could be convenient to synchronize the xPC Target application data and the measured test data. Since there is no direct access during the execution/real-time simulation of the xPC Target application of the ACS (control

algorithm) SIMULINK® model, the sample time of the optical measurement setup and the xPC system time could be initialized simultaneously.

Synchronized data would facilitate direct comparison between embedded ACS (control algorithm) SIMULINK® determined data and measured data.

V. CONCLUSION

A method to determine a location with small magnetic field deviation was developed and can be used for future measurements. Measurements of the actual magnetic field and the analysis of the data was successfully completed. The location of the air-bearing table was decided based on the results of this approach.

A LabVIEW^{TM} serial magnetometer driver was implemented for experimental purposes. It can be used as magnetometer configuration tool for measurements and it can easily be customized for other purposes. The work on a custom SIMULINK® implementation of a magnetometer driver was not finished. This SIMULINK® implementation is necessary for the use of the magnetometer as a measurement device on the air-bearing platform for hardware-in-the-loop simulation. The information that was obtained can be used to accomplish this work. The wireless data connection was established as a part of the work on these serial magnetometer drivers. This wireless setup is ready for use in hardware-in-the-loop simulations. Experience was gained in handling the xPC SIMULINK® software of the hardware-in-the-loop simulation. Different problems of the software setup were solved.

The setup of the air-bearing platform structure was finished during the work on NPSAT1 hardware-in-the-loop simulations. This air-bearing platform structure is the hardware model that will be controlled by the revised ACS SIMULINK® Model. Determining precisely or measuring the air-bearing platform mass properties will facilitate realistic simulation results of the ACS SIMULINK® and good conditions for hardware-in-the-loop simulation. Most of the electrical power supply system was finished. The air-bearing platform setup is ready for use after finishing the torque-rod driver board. Work on the optical measurement setup is nearly finished. The measurement software is ready for use. A remaining question is the synchronization

of both the measured data and the data of the real-time application of the embedded ACS SIMULINK®.

APPENDIX A. HMR2300 SPECIFICATIONS

The tables on the following pages are taken from the HMR2300 manual [Ref. 9]. They give a short overview of the most important specifications. However refer to that manual for further or more specific information.

Characteristic	*Conditions*	*Min*	*Typ*	*Max*	*Unit*
Field Range	FS - total applied Field	-2		$+2$	Gauss
Accuracy	RSS of all Errors at 25°Celsius $\pm$1Gauss $\pm$2Gauss		 0.12 1	 0.52 2	 %FS %FS
Resolution	applied Field to change Output	67			μGauss
max. exposed Field	no perming Effect on Zero reading			10	Gauss
Dimensions	$l \times b \times h$	$82.6 \times 38.1 \times 22.3$			mm

Table XIV. General Smart Digital Magnetometer HMR2300 Specifications

Field Value Gauss	*BCD ASCII counts*	*Binary Value* (Hex) High Byte	Low Byte
+2.0	30,000	75	30
+1.0	15,000	57	E4
+0.5	7,000	1D	4C
0.0	00	00	00
-0.5	-7,500	E2	B4
-1.0	-15,000	C3	74
-2.0	-30,000	8A	D0
ASCII Format, 28 bytes		$SN\|\xi_1\|\xi_2\|CM\|\xi_3\|\xi_4\|\xi_5\|SP\|SP\|$ $SN\|\eta_1\|\eta_2\|CM\|\eta_3\|\eta_4\|\eta_5\|SP\|SP\|$ $SN\|\zeta_1\|\zeta_2\|CM\|\zeta_3\|\zeta_4\|\zeta_5\|SP\|SP\| < cr >^{(1)}$	
Binary Format, 7 bytes		$\xi_h\|\xi_l\|\eta_h\|\eta_l\|\zeta_h\|\zeta_l\| < cr >^{(2)}$	

(1) SN = sign, SP = space; CM = comma; $\xi_1,\ \xi_2,\ \xi_3,\ \xi_4,\ \xi_5$ = decimal equivalent ASCII digits; $\xi_1,\ \xi_2,\ \xi_3$ = SP if leading digits are zero.

(2) ξ_h = signed high byte, ξ_l = low byte.

Table XV. HMR2300 Output Formats.

Command	*Inputs*[1]	*Response*	*Bytes*[2]	*Description*
Write Enable	*ddWE< *cr* >	OK¬	3	Required before all shown commands except output.
Default Settings	*ddWE< *cr* > *ddD< *cr* >	 OK¬ BAUD=_9600¬	14	Change all command parameters to factory default values[3].
Device ID	*99ID< *cr* > *ddWE< *cr* > *ddID=nn< *cr* >	ID=_nn¬ OK¬	7 3	Read the current device ID. Set device ID, $00 \leq nn \leq 98$.
Format	*ddWE< *cr* > *ddA< *cr* > *ddWE< *cr* > *ddB< *cr* >	 ASCII_ON¬ BINARY_ON¬	9 10	Output readings in BCD ASCII. Output readings in signed 16 bit.
Baud Rate	*99WE< *cr* > *99!BR=S< *cr* > *99WE< *cr* > *99!BR=F< *cr* >	 OK¬ BAUD=_9600¬ OK¬ BAUD=_19,200¬	14 14	Set baude rate to 9600*bps*. Set baude rate to 19200*bps*.
Sample Rate	*ddWE< *cr* > *ddR=nnn< *cr* >	 OK¬	3	Set sample rate to nnn[4].
Output	*ddP< *cr* > *ddC< *cr* >	$\xi,\ \eta,\ \zeta$ read. $\xi,\ \eta,\ \zeta$ stream	7 / 28 ...	Output a single sample. Output readings sample rate.

Table XVI. HMR2300 In- and Output Specifications.

(1) The "$< cr >$" carriage return, or Enter, is required after all commands. The symbol "¬" in the *Output* column is the carriage return of the output. The symbol "dd" means device ID. 99 is global address for all magnetometer units.

(2) This is the number of output bytes.

(3) Default values are: ASCII, single sample output, device ID = 00, baud rate = 9600bps, sample rate = 20 sps.

(4) The following sample rates can be chosen: nnn=10, 20, 25, 30, 40, 50, 60, 100, 123, 154 samples/sec (sps).

Characteristic	*Conditions*	*Min*	*Typ*	*Max*	*Unit*
response time (command - response)	*dd command *ddP *ddC *99 command	1.9	2 3 40 2 = dd*40	2.2 3.2 60 2 + Typ	msec
time delay (response of the dd device in a queue)	*dd command *99 command	39	40 dd*40	41 2+ Typ	msec

Table XVII. HMR2300 Timing Specifications.

APPENDIX B. MAGNETIC FIELD DATA ANALYSIS

This appendix contains Maple 7.0 work sheets or rather sequences of Maple 7.0 work sheets that were used for the analysis of measured magnetic field data. It contains also some information about the raw data.

RAW DATA

File Names

The measurements were taken with hardware and software that is described in II.B.2 and II.B.3. These data are logged with the HMR2300 demo software, [Ref. 10], into data files. These data files were named with the number `i` of that grid point at which the corresponding file was created. As the measurement setup facilitates measurements in three planes on the grid (see II.D.1 and II.D.2) these files were named consecutively. The grid was numbered lexicographical. Measurement planes and corresponding grid point numbers can be seen in Table II. This numbering allows easy automatized data import with Maple 7.0 and therefor the use of Maple 7.0's extensive statistical command library. Besides this there has not been any necessary formatting of raw data with other software like C or Excel.

File Format

The HMR2300 demo software writes data to a file in the format shown in Table XVIII.

Notice. This written data format is different from that one, specified for displayed output, see Table XV. As one can see in this sample, commas and spaces are used to separate columns.

↓ column [i][j]				
3217,	2955,	-2239,	317.4	
3217,	2954,	-2239,	317.4	
3217,	2954,	-2239,	317.4	← row
3217,	2954,	-2239,	317.4	
3217,	2952,	-2239,	317.5	
1	2	3		index j
ξ	η	ζ	*mag. north direct.*	
	[ ] = *counts*		[ ] = *degree*	

Table XVIII. Data File Format.

MAPLE 7.0 SEQUENCES

This section shows and explains Maple 7.0 work sheets that were used. The function of important inputs and commands is described briefly so that somebody who is not familiar with Maple can use these scripts. This may be necessary because the actual position of the air-bearing table is not final. These tools are offered to facilitate a fast determination of the new position.

The following subsections cover Maple 7.0 sequences that are mostly modules that can be implemented in only one Maple 7.0 work sheet. Different Maple 7.0 work sheets were used, each for another purpose. This was done to keep an overview and to have immediate access to results. Each of these particular sheets had to have its own initialization and data import sequence (see Paragraphs *B. Maple 7.0 Sequences. Initialization* and *B. Maple 7.0 Sequences. Import of Data Files*)

Maple 7.0 is a good calculation tool and convenient also for new users. Its online help function is very useful. The following approach to access that help can be useful. The Help button at the Maple 7.0 task bar menu facilitates access to the Glossary menu. One can find inhere many topics e.g. Mathematics... . One of

the submenus of Mathematics... is e.g. Packages.... There are shown all available topic specific command packages. Typing e.g. command `with(stats);` includes the package `stats` and results in the output shown below. Maple 7.0 displays the commands that are provided by including one package. The output after an executed command can be suppressed with the colon ":" ore displayed with the semicolon ";". Either the first or the second one is necessary to end and execute a command.

```
> restart;with(stats);
```

[*anova*, *describe*, *fit*, *importdata*, *random*, *statevalf*, *statplots*, *transform*]

```
> ?importdata
```

One Maple execution group is specified with ">". One can see in the second execution group the `?importdata` command. This is one way to access the Maple 7.0 online help. If one wants to know the use or syntax of a command one can simply type the question mark "`?`" in front of the command. Executing this command will display a command corresponding help. The execution of a command is suppressed by typing "`#`" in front of it. Both "`?`" and "`#`" does not necessitate the ":" or ";".

Initialization

The following execution group is used to initialize one Maple 7.0 work sheet. The packages `stats` and `describe` provide commands for statistical analysis, the first e.g. `importdata` and the second e.g. `mean` for determination of the empirical mean. Package `linalg` provides commands for vector and matrix calculus, e.g. `crossprod` to determine the cross product of two vectors. Package `plots` facilitates to present data in graphics, diagrams etc.

```
> restart:with(stats):with(describe):with(linalg):with(plots):
```

Warning, the protected names norm and trace have been redefined
and unprotected

Warning, the name changecoords has been redefined

Import of Data Files

The next execution group loads the data files of one magnetic field measurement into the Maple 7.0 work sheet. In this case are loaded `i`= 189 data files from one folder. The folder's path name is `c:/daten/06feb/dat`. This can take a few seconds.

The `cat(.)` command concatenates path name with loop index value `i`. This string is allocated to variable `n[i]` and passed to the `importdata` command. This command loads each column (see Table XVIII) in one data column of data `X[i][j]`, $1 \leq$ `i` $\leq 189,\ 1 \leq$ `j` ≤ 4. Only the first three columns, $1 \leq$ `j` ≤ 3, are used for this analysis. As one can see in Table XVIII, the last one contains the magnetic north direction that is not necessary for this analysis. However the `importdata` command has to read in four columns to provide the correct data in one column. Variable `N[i]` contains the number of values in each column `X[i][j]` determined with the `count` command. The columns are measured simultaneously and have therefor the same dimension.

```
> for i from 1 to 189 do
n[i]:=cat("c:/daten/06feb/dat/",i ):
X[i]:=[importdata(n[i],4)]:
N[i]:=count(X[i][1]):
od:
```

Scale Range, Disturbed Measurements

The following sequence is to check whether the scale range has been ±1Gauss or ±2Gauss during the measurements. Variables `s_typ` and `s_max` contain the corresponding accuracy values in *counts*, see Table IV. `k` is in this case the location index of a particular value `X[i][j][k]` in one column `X[i][j]`. Index `j` specifies the column, $1 \leq$ `j` ≤ 3. Index `i` specifies the grid point. In `x[i][k]` is contained the "Pythagoras" (norm) of one row of measured vector component values, see Table XVIII. These values are filled in a list (`seq` command) and their maximum is determined (`max`). This value is compared with the ±1Gauss *counts* equivalent.

```
> s_typ:=0.12*1/100*15000;

s_max:=0.52*1/100*15000;
> for i from 1 to 189 do

for k from 1 to N[i] do

x[i][k]:=sqrt(sum(X[i][j][k]^2,'j'=1..3)):

od:

liste_x:=seq(x[i][k],k=1..N[i]):

max_read[i]:=max(liste_x):

if (max_read[i] > 15000) then

printf( "Pt nr %d range +/- 2Gauss.\n",i):

end if;

od:
```

The upper part of the execution group below is to determine the empirical standard deviation `s[i][j]` of one column `X[i][j]` (`standarddeviation[1]` command.) It also compares these values with accuracy values (`if` case). If there would be a standard deviation bigger than `s_max` it should be an indicator for a disturbed

measurement. Because if the magnetic environment does not change, the variety or deviation of values in one column should be caused only by inaccuracy as result of statistical errors of the measurement device, not by outer magnetic influences. Conversely, if the standard deviation is smaller than `s_max` (the maximal allowance of deviation) one cannot decide whether the deviation of these values is caused by statistical errors or outer magnetic influences.

```
> for i from 1 to 189 do
for j from 1 to 3 do
s[i][j]:=standarddeviation[1](X[i][j]):
if (s[i][j] >= s_typ) then
printf( "s[%d][%d] >= s_typ.\n",i,j):
end if;
if (s[i][j] >= s_max) then
printf( "s[%d][%d] >= s_max.\n",i,j):
end if;
z[i][j]:=1:
a:=sort(X[i][j]);
for k from 1 to N[i]-1 do
if (a[k+1] > a[k]) then
z[i][j]:= z[i][j]+1:
end if:
od:
od:
od:
```

```
> liste_z:=seq(seq(z[a][b],b=1..3),a=1..189):
max_z:=max(liste_z);
```

The lower part of this sequence shown above determines also the number of different reading values ([ ] = *counts*) in one column `X[i][j]`. `z[i][j]` is the class counter of one column. The column is sorted (`sort` command) and stored in variable `a`. The `if` case compares successively two adjacent values and increments the class counter by one for each step in the sorted list of values `a`. After that is determined the maximum of all class counters for one measurement.

The execution group shown below delivers indices `i` and `j` that specify the samples `X[i][j]` with a maximal number of different values. It compares the maximal class counter `z_max` with all genuine data columns and displays their indices.

```
> for i from 1 to 189 do
for j from 1 to 3 do
if (z[i][j] = max_z) then
printf("X[%d][%d]-> max_z =%d.\n",i,j,max_z):
end if:
od:
od:
```

Empiric Distribution Function

The following sequence reads in data of just one data file (in this case `c:/daten/23jan/dat/128`) and classifies one sample or column `X[128][3]`. These data files are picked by another sequence, see Paragraph *B. Maple 7.0 Sequences. Scale Range, Disturbed Measurements*. The column is sorted and stored in `x`. Furthermore are initialized variable `z` as counter for the number of classes, Variable `iu` that indicates the first value of the current class and variable `io` that indicates the last value of the current class.

```
> restart;with(stats):with(describe):with(plots):
> X:=importdata("c:/daten/23jan/dat/128",4):
> N:=count(X[3]);
> x:=sort(X[3]):
> z:=1:iu:=0:io:=0:
```

The next `for` loop classifies the column. `k` is again the index that locates the position of one value within the sample. The first embedded `if` case looks for steps in the sorted column `x`. It sets `iu` to the index of the first value of the current class (this is the value after the last one in the class before) and `io` to the index of the last value in the current class. The current class is stored in `kl[z]` and the class counter is incremented by 1. `n[z]` is the dimension of the current class. The second embedded `if` case handles the last value if the first one does not. The `print(z)` command is for a manual check. The class counter `z` has to have the same value as `max_z` in Paragraph *B.Maple 7.0 Sequences.Scale Range, Disturbed Measurements*.

```
> for k from 1 to N-1 do
if (x[k] < x[k+1])then
iu:=io+1:
io:=k:
kl[z]:=[seq(x[i],i=iu..k)]:
n[z]:=count(kl[z]):
z:=z+1:
end if;
if (k=N-1) then
kl[z]:=[seq(x[i],i=io+1..N)]:
n[z]:=count(kl[z]):
end if:
od:
> print(z);
```

The next execution group is to check manually for lost values. This can be done just with nominal/actual comparison of sample dimension `N` with the cumulative number of values in classes `sum(n[i],'i'=1..z)`.

```
> sum(n[i],'i'=1..z);
```

This execution group below determines the empiric distribution function `F[j]`, see Equation II.5. `j` is in this case the class index, $1 \leq$ `j` $\leq$ `z`. It also delivers all data points `P[j]` to plot the empiric distribution function. `kl[j][1]` is just the first value of the current class, used to represent it.

```
> for j from 1 to z do
F[j]:=1/N*sum(n[i],'i'=1..j):
P[j]:=[kl[j][1],F[j]];
od:
```

The two execution groups that follow are just to create the plot structure and to display the graph of the empiric distribution function.

```
> p:=[seq(P[j],j=1..z)];
> pointplot(p,style=line,labels=["x[z]","F(x[z])"]);
```

Mean, Standard Deviation, C.I.

The Maple 7.0 sequences described before were mostly preliminary considerations.

The sequence shown below determines all values that are necessary to describe one sample `X[i][j]`. `j` is again the index that specifies the column at a particular grid point `i`. `alpha` is the chosen probability of error. `t[i]` is the quantile of Student's distribution determined with the `statevalf[icdf,...]` command. With `xquer[i]` is created a zero vector (`vector` command). The embedded second `for` loop fills the components of `xquer[i]` with mean values of columns `X[i][j]`. These mean values are the components of the magnetic flux density vector in unit *counts* (see Table XV and [Ref. 9]) respectively the magnetometer body coordinate system, see Figure 3. [`xu[i][j]`,`xo[i][j]`] are the range of the confidence interval C.I., see Equation II.9.

```
> alpha:=0.05:
> for i from 1 to 189 do

t[i]:=statevalf[icdf,studentst[N[i]-1]](1-alpha/2):

xquer[i]:=vector(3,[seq(0,i=1..3)]):

for j from 1 to 3 do

xquer[i][j]:=mean(X[i][j]):

s[i][j]:=standarddeviation[1](X[i][j]):

xu[i][j]:=xquer[i][j]-t[i]*s[i][j]/sqrt(N[i]):

xo[i][j]:=xquer[i][j]+t[i]*s[i][j]/sqrt(N[i]):

od:

od:
```

Vector Field Plot

The next execution group scales the corresponding mean values `xquer[i][j]`, $1 \leq$ `j` ≤ 3, to fit with a drawing grid. It combines them in `B[i]` magnetic flux density field vectors conform to the coordinate transformation stated in Equation II.3. These vectors are scaled to fit with a drawing grid for a convenient visualization.

`k` is in this case the index of grid rows, $0 \leq$ `k` ≤ 8 (see Figure 27). `j` is the measurement plane index, $0 \leq$ `j` ≤ 2. `1+k*7` specifies therefore the first point of a row, `7+k*7` the last one. `j*63` shifts to the corresponding measurement plane, similar to the term `k*7` that shifts rows, see also Figure 27.

`u[i]` are the position vectors that point at the `i`th grid point. The `arrow` commands create a plot structure and draw each vector `B[i]` at the position `u[i]`.

```
> for i from 1 to 189 do
B[i]:=[-xquer[i]][3]*1/150,xquer[i][2]*1/150,xquer[i][1]*1/150];
for k from 0 to 8 do
for j from 0 to 2 do
if (i >= (k*7+1+j*63)) and (i <= (k*7+7+j*63)) then
u[i]:=[k*50, (i-(k*7+1+j*63))*50, (2-j)*50];
p[i]:=arrow(u[i],B[i]);
end if:
od:
od:
od:
```

The following sequence builds the plot structures for x and y drawing grid lines. It uses the same `arrow` command as above and is therefor a verification of a

correct graphical display of the `B[i]` vectors. If the grid would not look similar to Figure 2 the display of magnetic field vectors would be wrong, too. The `display` command causes finally graphical output.

```
> for k from 0 to 8 do
for j from 0 to 2 do
gy[k][j]:=arrow([k*50,0,j*50],[0,6*50,0],shape=harpoon,
head_width=0,color=black):
if (k<=6) then
gx[k][j]:=arrow([0,k*50,j*50],[8*50,0,0],shape=harpoon,
head_width=0,color=black):
end if:
od:
od:
> p[0]:=arrow([0,0,0],labels=['x','y','z']):
> display3d(seq(p[i],i=0..189),
seq(seq(gx[k][j],k=0..6),j=0..2),
seq(seq(gy[k][j],k=0..8),j=0..2));
```

k=8	57	58	59	60	61	62	63	
k=7	50	51	52	53	54	55	56	
k=6	43	44	45	46	47	48	49	
k=5	36	37	38	39	40	41	42	
k=4	29	30	31	32	33	34	35	
k=3	22	23	24	25	26	27	28	
k=2	15	16	17	18	19	20	21	
k=1	8	9	10	11	12	13	14	
k=0	1	2	3	4	5	6	7	i=(1+k·7)..(7+k·7)

Figure 27. Field Indices I.

k=6		51	52	53	54	55	
k=5		44	45	46	47	48	
k=4		37	38	39	40	41	
k=3		30	31	32	33	34	
k=2		23	24	25	26	27	
k=1		16	17	18	19	20	
k=0		9	10	11	12	13	i=(9+k·7)..(13+k·7)

Figure 28. Field Indices II.

Median, Median Plot

The execution groups shown below are used to determine the median `xm[j]` (`median` command) of all ξ, η or ζ components, `xquer[i][1]`, `xquer[i][2]`, `xquer[i][3]`, $1 \leq$ `i` ≤ 189 (respectively 63). `farbe` is a structure that is used to code `xm[j]` with a color. `pointplot` commands build the plot structure of data points [`i`,`xquer[i][j]`]. `plot` commands draw the median values as a line. `p[0]` is just to name the diagram axes.

```
> farbe:=[red,green,black];
> for j from 1 to 3 do

xm[j]:=median([seq(xquer[i][j],i=1..189)]):

p[j]:=pointplot([seq([i,xquer[i][j]],i=1..189)],color=farbe[j]):

p[j+3]:=plot(xm[j],x=0..63,color=farbe[j]):

od:

> p[0]:=plot(0,x=0..63,color=black, labels=["i","counts"]):

> display(seq(p[i],i=0..6));
```

Difference Vectors, Sum of spanned Areas

The sequence below has initialized the Maple 7.0 work sheet that was used finally to determine the position of the air-bearing platform. The data are read in like it is described before. The `vector` command builds a zero vector. The three corresponding mean values `xquer[i][j]`, $1 \leq$ `j` ≤ 3 at one particular grid point `i` are filled in this vector. This facilitates the use of commands of the `linalg` package.

```
> restart:with(stats):with(describe):with(linalg):with(plots):
> for i from 1 to 189 do

n[i]:=cat( "c:/daten/23jan/dat/",i ):

X[i]:=[importdata(n[i],4)]:

xquer[i]:=vector(3,[seq(0,i=1..3)]):

for j from 1 to 3 do

xquer[i][j]:=mean(X[i][j]):

od:

od:
```

The next execution group calculates the scaled difference vectors `r[i][j]`, `j` $\in \{$`i` $-7,$ `i` $-1,$ `i` $+1,$ `i` $+7\}$, see also Figure 10. `k` is now the index of the inner seven rows of the grid and `i` selects the inner grid points (see also Figure 11). `9+k*7` is the first value of an inner row and `13+k*7` is the last one, see Figure 28. By adding again `j*63` to these values can be switched to another measurement plane. The `norm` command determines the norm of the central vector `xquer[i]`.

Both embedded `for` loops are used to change the indices $\{$`i` $-7,$ `i` $-1,$ `i` $+$ $1,$ `i`$+7\}$ cyclical. The `crossprod` command determines the vector cross product. This is used to calculate the triangular areas `a[i][l][m]` spanned between two adjacent

difference vectors. These little areas are added and stored in `A[0][i-(8+2*k)]`. Index `A[0]` indicates the lowest measurement plane. `[i-(8+2*k)]` changes the interrupted genuine grid point indexing to a new (artificial) successive numbering $[1, 2, \ldots, 35]$.

```
> for k from 0 to 6 do
for i from (9+k*7) to (13+k*7) do
normxquer[i]:=norm(xquer[i],'frobenius'):
r[i][i+1]:=evalm(xquer[i+1]-xquer[i])/normxquer[i]:
r[i][i+7]:=evalm(xquer[i+7]-xquer[i])/normxquer[i]:
r[i][i-1]:=evalm(xquer[i-1]-xquer[i])/normxquer[i]:
r[i][i-7]:=evalm(xquer[i-7]-xquer[i])/normxquer[i]:
for l from 1 to 2 do
for m from 1 to 2 do
a[i][l][m]:=0.5*norm(crossprod(r[i][i+(-1)^l],r[i][i+7*(-1)^m]),'frobe
nius'):
od:
od:
A[0][i-(8+2*k)]:=a[i][1][1]+a[i][1][2]+a[i][2][1]+a[i][2][2];
od:
od:
```

The last two execution groups of this work sheet are to pick the five smallest `A[0]` values of the spanned areas. This is an advanced version of the sequence that looks for the samples with most different values in it, see Paragraph *B. Maple 7.0 Sequences. Scale Range, Disturbed Measurement.*

```
>  liste_A[0]:=seq(A[0][j],j=1..35):

sort_A[0]:=sort([liste_A[0]]);
>  for i from 1 to 35 do

for j from 1 to 5 do

if (A[0][i] = sort_A[0][j]) then

printf("A[0][%d] = %f.\n",i,sort_A[0][j]):

end if:

od:

od:
```

MAPLE 7.0 PLOTS

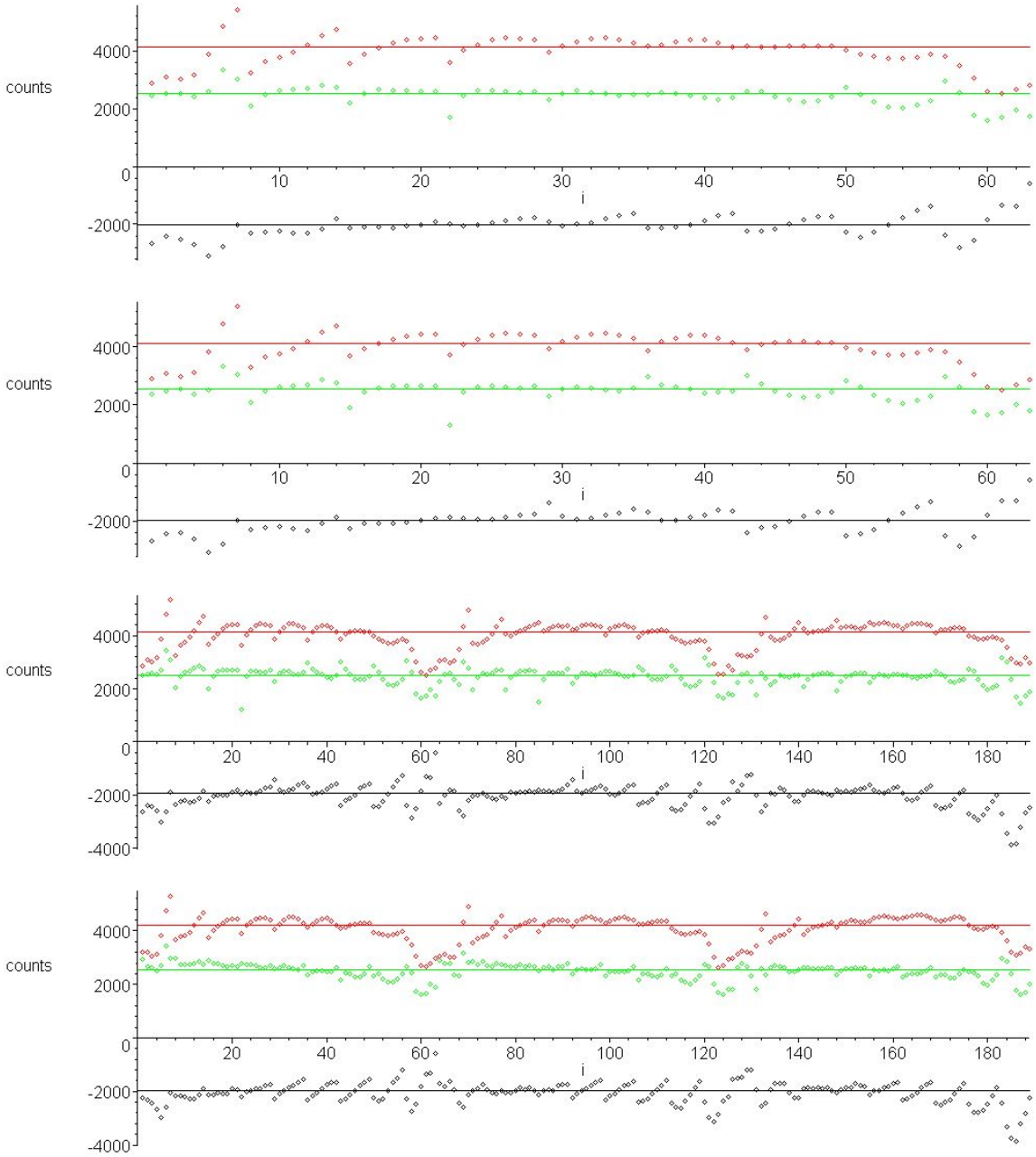
In this section are placed some visualizations that were obtained during the process of analyzing the measured magnetic field data.

Median Plots

These data point plots show at each grid point i all three components of the measured magnetic flux density vector $\mathbf{B}_i$. The unit of these components is *counts*, that is a HMR2300 magnetometer specific unit. These shown component point plots represent the mean of the corresponding sample measured at the i-th grid point. Both measurements from 10th Jan and 17th Jan were taken only in one plane.

One can see in these four diagrams in Figure 29 that there are two systematical effects. One can be found in each single measurement row (refer to Figure 27 for row indices). The other one can be found in each measurement plane (both lower diagrams in Figure 29). The first systematical effect is the change of ξ, η, and ζ components (respectively the magnetometer coordinate system) in each grid row (seven successive grid points). Interesting is that these changes are not the same in each row and for each direction. There are less changes at the inner measured field sections. The sources of these changes does not affect all three components in the same way. It seems that ξ and ζ components changes in one row as well as in one plane more than η. The second systematical effect can be seen in each plane (refer to Table II for plane indices). There are rows that are as a whole closer to the corresponding median than other ones. The sources of these changes seem not to affect each plane in the same way. The higher the plane the less bow in one row, compared to its corresponding lower ones.

The Interpretation of these effects is that there are structural building elements beneath the floor as well as other sources (e.g Uninterruptible Power Supplies) that have created fields superimposed with the Earth's magnetic field. An indicator for that is the second systematical effect that depends on the measurement hight.



ξ readings:	red diamonds,	ξ median:	red line
η readings:	green diamonds,	η median:	green line
ζ readings:	black diamonds,	ζ median:	black line

top: 10th Jan mid. top: 17th Jan mid. bot.: 23rd Jan bot.: 06th Feb

Figure 29. Median Plots.

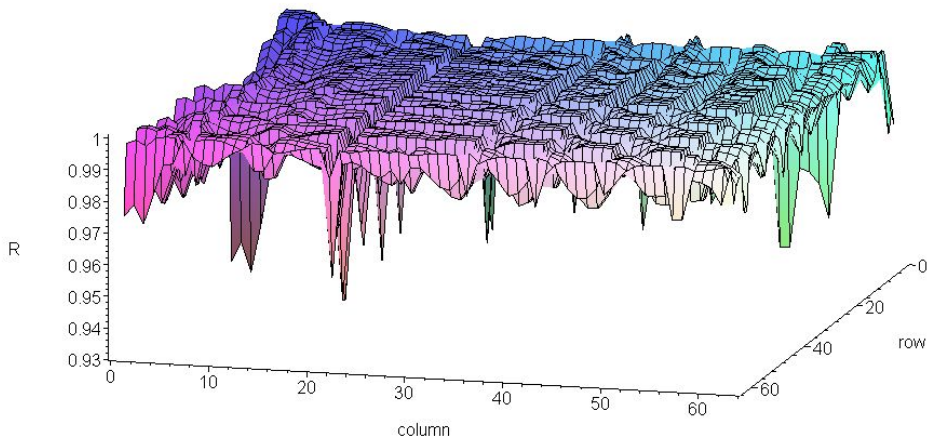


Figure 30. Vector Correlation Coefficients Plot.

Correlation Coefficients Plot

In Paragraph II.D.3 is mentioned that correlation coefficients between field vectors were considered as measure for magnetic field homogeneity. There are stated also good reasons to avoid this approach. A small Maple 7.0 work sheet of this approach was implemented. This Maple 7.0 work sheet is also part of the data CD.

The measurement of Jan 17 were taken as example and the correlation coefficients of all vectors $\mathbf{B}_i$ were determined. These vectors were composed from the means $\bar{x}_{i,j}$ of the measured samples $\mathbf{X}_{i,j}$, $i \in \{1, 63\}$, $j \in \{\xi,\ \eta,\ \zeta\}$. The result is visualized in Figure 30.

One can see in Figure 30 the correlation coefficient displayed as height above an 63×63 grid. The correlation coefficient of e.g. vectors $\mathbf{B}_{22}$ and $\mathbf{B}_{45}$ is located at position $22, 45$ and $45, 22$ because the matrix $\mathbf{R}$ of correlation coefficients is symmetric. One can find similarities between Figure 30 and Figure 29. Both specify the field vector $\mathbf{B_{22}}$ as "outlier". But one can see in Figure 29 that vector $\mathbf{B}_7$ does not match either. This is not clearly indicated in Figure 30. The range of correlation coefficients

(see scale in Figure 30) is not that wide that there can be found an appropriate measure to define which vectors are outliers. If one takes a look at Figure 7, one can find those problems confirmed that are mentioned in Paragraph II.D.3.

APPENDIX C. SERIAL EXPERIMENT SIMULINK® MODELS.

This appendix contains information of those SIMULINK® models that were used for various attempts to establish correct serial communication between an xPC Target application and the HMR2300 magnetometer.

XPC RS232 ASYNCHRONOUS EXPERIMENTS

The very simple xPC/SIMULINK® asynchronous RS232 experimental driver model can be see in Figure 31. It can be implemented with blocks from the SIMULINK® library browser, sections xPC, Sources and Sinks. How to implement a xPC/SIMULINK® model can be seen in [Ref. 20]. For explanations of SIMULINK® blocks refer to the SIMULINK® online context help that can be accessed with right mouse click on a block and choosing the help submenu.

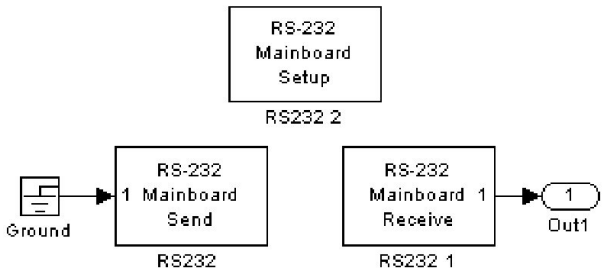


Figure 31. xPC/SIMULINK® RS232 Async Experimental Model.

RS232 Setup Block

This block is described briefly in Paragraph III.D.1. The RS232 Setup block parameter dialog box is shown in Figure 32. Refer to [Ref. 22] for very short explanations of each input line of the RS232-Setup block dialog box. In Table XIX are shown briefly only the necessary ones.

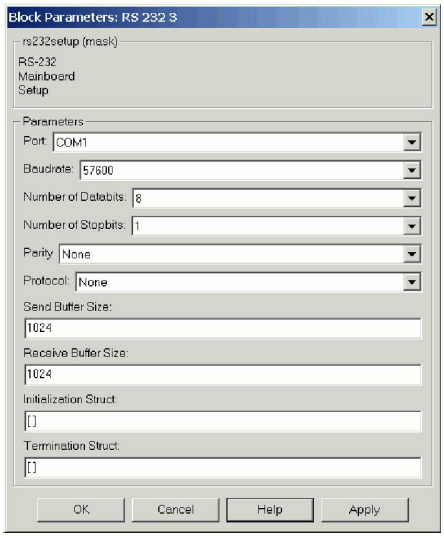


Figure 32. RS232-Setup Block Parameters Dialog Box.

Parameter	*Setting*	*Comment*
Port	COM1	has to match to RS232 Send/Receive block settings
BaudRate	9600	has to match HMR2300 properties
all others		default settings were used

Table XIX. RS232-Setup Block Parameters.

RS232 Asynchronous Send Block

This block is described briefly in Paragraph III.D.1, too. The RS232 Send block parameter dialog box is shown in Figure 33. Refer to [Ref. 22] for very short explanations of each input line of the RS232-Setup block dialog box.

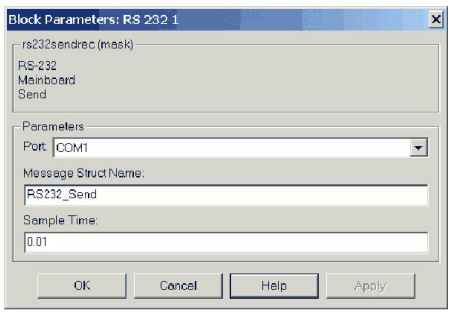


Figure 33. RS232-Send Block Parameters Dialog Box.

Parameter	*Setting*	*Comment*
Port	COM1	the COM port used for sending data to the serial device
Message Struct Name	RS232_Send	created by Message Structure M-files, see also [Ref. 22]
Sample Time	1	

Table XX. RS232-Send Block Parameters.

RS232 Asynchronous Receive Block

This block is described briefly also in Paragraph III.D.1. The RS232 Receive block parameter dialog box is shown in Figure 34. Refer to [Ref. 22] for very short explanations of each input line of the RS232-Setup block dialog box.

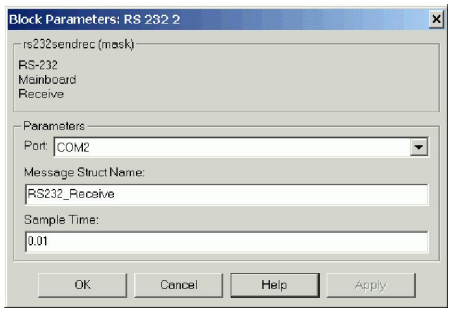


Figure 34. RS232-Receive Block Parameters Dialog Box.

Parameter	*Setting*	*Comment*
Port	COM1	the COM port used for receiving data, usually the same as in corresponding RS232 Send block specified
Message Struct Name	RS232_Receive	created by Message Structure M-files, see also [Ref. 22]
Sample Time	1	to match the corresponding RS232 Send block

Table XXI. RS232-Receive Block Parameters.

RS232 Asynchronous Message Structures

An example of these message structures is shown in Table XXII. They have to be implemented in a MATLAB® M-file and loaded into MATLAB® workspace, see [Ref. 22]. The SIMULINK® model can be updated after these message structures are loaded. That means the model has to be executed once. There appear in- and output connectors at the RS232 asynchronous send/receive blocks after these message structures are loaded.

MATLAB® M-File Message Structure	*Description*
`RS232_Send(1).SendData = '*00#\r' ;`	, command to be sent
`RS232_Send(1).InputPorts = [1];`	, number (names) of send ports
`RS232_Send(1).Timout = 0.01;`	, time to wait for returned data
`RS232_Send(1).EOM = 0;`	, numbers of characters that indicates the end of a message
`RS232_Receive(1).RecData = 'SER\# %u \r';`	, expected answer
`RS232_Receive(1).OutputPorts = [1];`	, number of receive ports
`RS232_Receive(1).Timout = 0.01;`	, time to wait for returned data
`RS232_Receive(1).EOM = 1;`	, numbers of characters that indicates the end of a message

Table XXII. RS232 Async Message Structures.

XPC RS232 ASYNC/BIN EXPERIMENTS

Figure 35 shows the xPC/SIMULINK® RS232 asynchronous/binary experimental model. Blocks RS232 Mainboard Setup, RS232 Mainboard Send, Unpack and RS232 Receive COM1 can be found at the SIMULINK® library browser, section xPC. The black bar is the Mux block from section SIMULINK, subsection Signal Routing. This model combines text based commands and binary responses. The message structure and block settings follow.

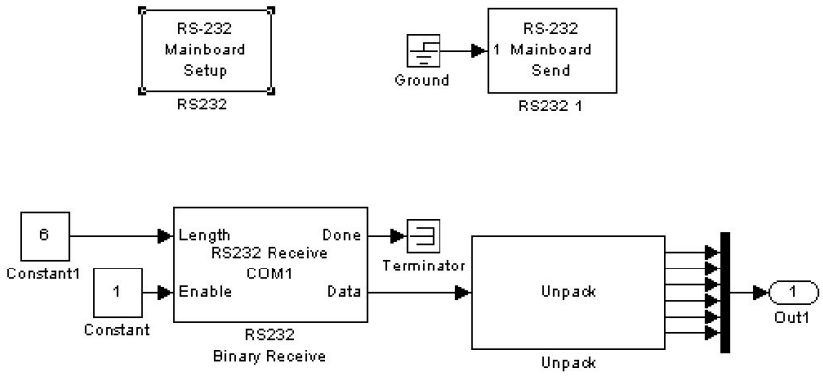


Figure 35. xPC RS232 Async/Bin Experimental SIMULINK® Model.

RS232 Binary Receive Block

A short description of that block can be found in Paragraph III.D.1. The RS232 binary receive block parameter dialog box is shown in Figure 36, [Ref. 22]. Its settings are contained in Table XXIII.

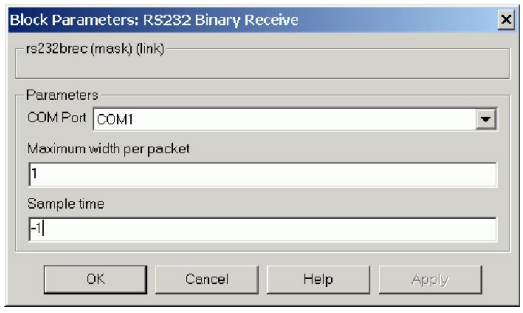


Figure 36. RS2323 Binary Receive Block Parameters Dialog Box.

Parameter	*Setting*	*Comment*
COM Port	COM1	the COM port used for receiving data, usually the same as in the corresponding RS232 Send block specified
Maximum Width per Packet	6	maximum byte length of the received data and width of the block output vector
Sample Time	1	to match the corresponding RS232 Send block

Table XXIII. RS232 Binary Receive Block Parameters.

Unpack Block

Also the Unpack block is described very briefly in Paragraph III.D.1 and [Ref. 23]. Its block parameter dialog box is shown in Figure 37 and the corresponding settings can be seen in Table XXIV.

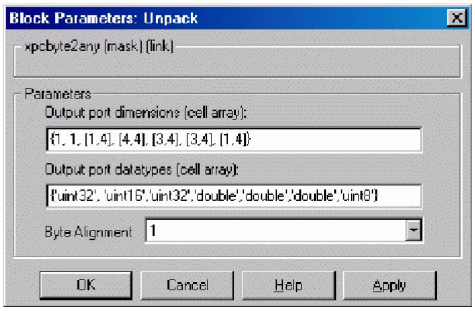


Figure 37. Unpack Block Parameters Dialog Box.

Parameter	*Setting*	*Comment*
Output Port Dimensions (Cell Array)	{1,1,1,1,1,1}	the structure of the received data, in this case 6 single values
Output Port Data-types (Cell Array)	{uint8,uint8,uint8, uint8,uint8,uint8}	uint8 is a supported data type that matches HMR2300 binary output, Table XV, these values represent those, in Output Port Dimensions specified
Sample Time	1	to match the corresponding RS232 Binary Receive block

Table XXIV. Unpack Block Parameters.

RS232 Asynchronous/Binary Message Structures

.

These message structures are quite similar to the RS232 asynchronous message structures. The only difference is that `RS232_Receive( ).` commands are missing. They are replaced with RS232 Binary Receive and Unpack block. The message structure and its short description can be seen in Table XXV. Further information can be found in [Ref. 22]

MATLAB® M-File Message Structure	*Description*
`RS232_Send(1).SendData = '*00#\r' ;`	, command to be sent
`RS232_Send(1).InputPorts = [1];`	, numbers (names) of send ports
`RS232_Send(1).Timout = 0.01;`	, time to wait for returned data
`RS232_Send(1).EOM = 0;`	, numbers of characters that indicates the end of a message

Table XXV. RS232 Async/Bin Message Structures.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D. OVERVIEW OF TECHNICAL DRAWINGS

The next three Tables, Table XXVI, Table XXVII and Table XXVII contain an overview of all technical drawings that were created to manufacture the parts of the air-bearing platform. The latest revisions of the technical drawings are contained at the data CD of this report.

Component	*Drawing (File Name)*	*Comment*
assembly	assytop	assembly drawing, top view and parts list
	assyfront	assembly drawing, front view
	assyside	assembly drawing, side view
angle-cross	angleA	upright part that supports the ζ magnetic torque rod and its three dummy rods
	angleB	upright part that supports the ζ magnetic torque rod and its three dummy rods, correspondent mirror part of angleA
	anglespacer	to connect angles, to establish gap for moving cross-members
	bottomspacer	to connect angles, to establish gap for moving cross-members, connects angles with disk
	cross	to connect angles with disk, bearing of spindle, to establish gap for moving cross-members
	cross-memberA	to connect torque rod and dummy rods that are mounted in ζ direction, lower part
	cross-memberB	connects torque rod and dummy rods that are mounted in ζ direction, upper part
	spindle	to connect both cross-members with cross, to move torque rod and dummy rods

Table XXVI. Overview of Technical Drawings I.

Component	*Drawing (File Name)*	*Comment*
basic plate	disk	to connect air-bearing inner (convex) with air bearing platform
	sexangle	to mounted to disk, supports most other parts, general drawing with over all dimensions
	sexangle2	drawing specifying the outer geometry
	sexangle3	drawing specifying dimensions of the hole pattern
	sexangle4	drawing specifying dimensions of the slot pattern
boxes: battery	epbox	covers EP battery, view from above
	epbox2	side view
	jbox1	covers J-Cell battery package, front view
	jbox2	view from above
	jbox3	side view
	box-thread	to mount battery boxes at sexangle plate
electrical power supply	base-plate eps	mounting device for electrical power supply box
target PC (CPU)	cpuangle	mounting device for CPU box on sexangle
	pin	mounting devices for boxes on sexangle
dummy rods:	dummyA	assembly, ζ dummy rod
	rod,br	inner part dummyA
	tube,al	outer part dummyA
	dummyB	assembly, ξ, η rods
	rod,al	inner part dummyB
	tube,br	outer part dummyB
	bracket	mounting device dummy rods

Table XXVII. Overview of Technical Drawings II.

Component	*Drawing (File Name)*	*Comment*
air-bearing lift mechanism	 lift-mechanism shell slope ring upper-slope-ring guide-ring top-ring lift-thread	 assembly connecting part with pedestal static part of mechanism, to support moving upper-slope ring moving part of mechanism to guide moving upper-slope ring to support air-bearing platform to connect upper-slope and ring top ring
magnetometer	mag-angle mag-plate	to mount magnetometer basic plate and magnetometer to sexangle plate magnetometer mounting plate

Table XXVIII. Overview of Technical Drawings III.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E. DETERMINATION OF ESTIMATED AIR-BEARING PLATFORM PROPERTIES

MASS PROPERTIES

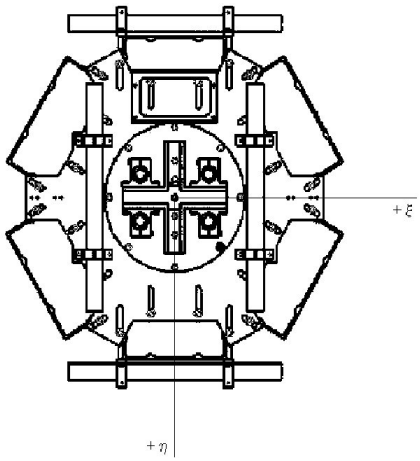
The mathematical model that was used to design the air-bearing platform can be used also to estimate some mass property values in the future phase to setup the air-bearing platform for hardware-in-the-loop simulations. Since not all components are finished can be used only estimated coordinates and masses. The air-bearing body coordinate system can be seen in Figure 38. All coordinates are based on this coordinate system.

Notice. To keep the work sheet readable, coordinates are named in the Maple 7.0 work sheet with their Latin expressions:

$$\begin{bmatrix} \xi \\ \eta \\ \zeta \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}. \tag{E.1}$$

These estimated mass property values are not to be used with the ACS (air-bearing) SIMULINK® model or ACS (control algorithm) SIMULINK® model. Some reasons for that are mentioned in Sections IV.A and IV.B. This Maple 7.0 work sheet may offer help in adjusting the real mass properties in a certain range before they are measured. So it may give a coarse reference for the influences of changed component positions to the air-bearing platform mass properties.

All following Maple 7.0 sequences are part of one work sheet and can be implemented in the shown order.



The ζ axis completes a right-hand coordinate system. The origin of the coordinate system is located at the center of rotation, refer to the air-bearing table technical drawings of Nelson Air Corp. ([Ref. 26]), available at SSAG.

Figure 38. Air-bearing Platform Coordinate System.

Initialization

The following execution group initializes the Maple 7.0 work sheet that determines the coordinates of the center of mass and the moments of inertia of the air-bearing platform. Indices `i` specify the components of the air-bearing platform. These are only the major components. Smaller supporting parts, screws, bolts and nuts are left out. The result would not have been worth the effort.

```
> restart;with(stats):with(describe):
#i=1,2-xrods, i=3,4-yrods, i=5..10-jcells, # i=11..14-zrods,
#i=15..18-angles, i=19-cpu_box, i=20-eps_box, i=21-magnetometer,
#i=22-wireless ethernet, i=23-disk, i=24-sexangle, i=25-bowl,
#i=26..29 ep_battaries
```

Cylindrical/Spherical Components

The next execution groups define vector `r` that is the vector of component radiuses and vector `l` that is the vector of corresponding component lengths. The `i`-th vector element is the property of the `i`-th air-bearing platform component, see Paragraph *E. Mass Properties. Initialization.*

Notice. The corresponding dimension of `r[i]`, `l[i]` is SI unit m. The actual value is specified in mm and converted to m with factor `1e-3`.

```
> r:=1e-3*[seq(14,i=1..4),
seq(26,i=5...10), seq(14,i=11..14), seq(0,i=15..22), 112, 0,
77, seq(0,i=26..29)];count(r);
> l:=1e-3*[seq(350,i=1..4),
seq(124,i=5...10), seq(350,i=11..14), seq(550,i=15..18),
seq(0,i=19..29)];count(l);
```

Cuboid Components

The following sequence defines vector `a`, vector `b` and vector `c`. These vectors contain the dimensions of cuboid air-bearing platform components. The `i`-th vector element corresponds to the `i`-th platform component, see Paragraph *E. Mass Properties. Initialization.*

Notice. The corresponding dimension of `a[i]`, `b[i]` and `c[i]` is SI unit m. The actual value is specified in mm and converted to m with factor `1e-3`. Length `a` is measured aligned with air-bearing platform axis ζ (respectively z). Length `b` is measured aligned with air-bearing platform axis η (respectively y) and length `c` is measured aligned with air-bearing platform axis ξ (respectively x). So these lengths depends on the orientation of the part respectively the air-bearing coordinate system.

```
> a:=1e-3*[seq(0,i=1..18), 228.6, 149.15,

82.6, 125.5, seq(0,i=23..25), seq(130,i=26..29)]; count(a);
> b:=1e-3*[seq(0,i=1..18), 50.2, 107.92,

38.1, 94, seq(0,i=23..25), seq(175,i=26..29)];count(b);
> c:=1e-3*[seq(0,i=1..18), 177.8, 76.18,

22.3, 31, seq(0,i=23..25), seq(84,i=26..29)];count(c);
```

Masses, Component Center of Mass Coordinates

The sequence shown below defines vector `m`, vector `xs`, `ys` and vector `zs`. Vector `m` contains the component masses. The `i`-th vector element corresponds to the `i`-th platform component, see Paragraph *E. Mass Properties. Initialization.* Variable `N` is the number of components. Each vector has to have this dimension, that is why each vector is checked with command `count`.

Notice. The corresponding dimension of `xs[i]`, `ys[i]` and `zs[i]` is SI unit m. The actual value is specified in mm and converted to m with factor `1e-3`. Coordinates `xs` are measured aligned with air-bearing platform axis ξ (respectively x). This applies analogously to the other coordinates. The corresponding dimension of `m[i]` is SI unit kg. The actual value is specified in kg. The center of mass coordinates are based on platform and component dimensions. These coordinates are the estimated distances of the center of mass of one part to the center of rotation measured aligned with the platform coordinate system. They can be determined by adding the location of the component center of mass within a part and the offset of that part. All components have been considered as basic geometrical objects with equally distributed mass (cuboid, cylinder, etc.). Their location of center of mass can be found in basic formularies, e.g. [Ref. 14].

```
> m:=[seq(1.05,i=1..4),
seq(0.84+0.375/3,i=5...10), seq(1.05,i=11..14),
seq(2.525*l[i],i=15..18), 0.850, 0.880, 0.094, 0.200, 1.356,
3.305, 8.75,
seq(4.9+0.5,i=26..29)];N:=count(m);
```

```
> xs:=1e-3*[seq(0,i=1..2), 137,-137,
52,0,-52, -52,0,52, -56,56,56,-56, -48,48,48,-48, 0, 0, 270, -260,
seq(0,i=23..25), -190,190,190,-190];
count(xs);
> ys:=1e-3*[265,-265, seq(0,i=3..4),
seq(210,i=5..7), seq(-210,i=8..10), 36,36,-36,-36, 48,48,-48,-48,
140, -155, seq(0,i=21..25), 114,114,-114,-114];
count(ys);
> zs:=1e-3*[121.7,121.7, -19,-19,
seq(89.7,i=5..10), seq(-192.7,i=11..14),
seq(-(l[i]*1e3/2+12.7),i=15..18), -110, -90, 85, 32 , -6.35,6.35,
29, seq(92.7,i=26..29)];
count(zs);
```

Platform Center of Mass Coordinates

The following execution groups determine the coordinates of the air-bearing platform center of mass respectively the platform coordinate system, see Figure kos. The `sum` command adds all masses `m[k]` that are specified within index range `1..N`. The center of mass coordinates are determined with the well-known relationship:

$$m_{result} \cdot x_{cm,result} = \sum_{k=1}^{N} m_k \cdot x_{cm,k}. \tag{E.2}$$

```
> M:=sum(m[k],'k'=1..N);
> Xs:=sum(m[k]*xs[k],'k'=1..N)/M;
> Ys:=sum(m[k]*ys[k],'k'=1..N)/M;
> Zs:=sum(m[k]*zs[k],'k'=1..N)/M;
```

Component Moments of Inertia

The next very long execution group determines particular moments of inertia. The different cases depends on whether the part is a cuboid, cylinder or from different geometry. These formulas uses the component properties that are specified in previously defined vectors `r`, `a`, `xs`, ..., see Paragraphs above.

Its current appearance has resulted from different changes on component specific moments of inertia. The `for` loop was used to automatize the determination of component moments of inertia since much of them have the same formulas but different dimensions. An appropriate order of that components has facilitated the use of the same formulas for a certain range of index `i` with properties specified as `i`-th property vector element. This has worked out for strict symmetrical design respectively moments of inertia. When moments of inertia have been specified more detailed (e.g. dummy rods) the few `if` cases had to be extended. I have neglect to fit the implemented work sheet structure to these new conditions. I think it is not that necessary.

```
> for i from 1 to N do
if (i=1) then
Ix[i]:=1/2*m[i]*r[i]^2              + (ys[i]^2 + zs[i]^2)*m[i];
Iy[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (zs[i]^2 + xs[i]^2)*m[i];
Iz[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (xs[i]^2 + ys[i]^2)*m[i];
end if;
```

...

```
   if (i=2) then
   Ix[i]:=121.891e-6     + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=10.718e-3      + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=10.718e-3      + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
   if (i=3) then
   Ix[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=1/2*m[i]*r[i]^2               + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
   if (i=4) then
   Ix[i]:=10.718e-3      + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=121.891e-6     + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=10.718e-3      + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
   if (i>=5) and (i<=11) then
   Ix[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=1/12*m[i]*(3*r[i]^2 + l[i]^2) + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=1/2*m[i]*r[i]^2               + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
...
```

```
   if (i>=12) and (i<=14) then
   Ix[i]:=10.6925e-3 + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=10.6925e-3 + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=69.4731e-6 + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
   if (i>=15) and (i<=18) then
   Ix[i]:=2.1e-3*l[i] + 0.842*l[i]^3  + (ys[i]^2 + zs[i]^2)*m[i];
   Iy[i]:=2.1e-3*l[i] + 0.842*l[i]^3  + (zs[i]^2 + xs[i]^2)*m[i];
   Iz[i]:=4.2e-3*l[i]                 + (xs[i]^2 + ys[i]^2)*m[i];
   end if;
   if (i>=19) and (i<=22) then
   Iz[i]:=1/12*m[i]*(b[i]^2 + c[i]^2) + (xs[i]^2 + ys[i]^2)*m[i];
   Iy[i]:=1/12*m[i]*(c[i]^2 + a[i]^2) + (zs[i]^2 + xs[i]^2)*m[i];
   Ix[i]:=1/12*m[i]*(a[i]^2 + b[i]^2) + (ys[i]^2 + zs[i]^2)*m[i];
   end if;
   if (i=23) then
   Ix[i]:=4.325e-3;
   Iy[i]:=Ix[i];
   Iz[i]:=8.504e-3;
   end if;
...
```

```
if (i=24) then
Ix[i]:=43.923e-3;
Iy[i]:=43.319e-3;
Iz[i]:=86.763e-3;
end if;
if (i=25) then
Ix[i]:=2/5*m[i]*r[i];
Iy[i]:=Ix[i];
Iz[i]:=Ix[i];
end if;
if (i>=26) and (i<=29) then
Ix[i]:=18.573e-3                    + (ys[i]^2 + zs[i]^2)*m[i];
Iy[i]:=13.593e-3                    + (zs[i]^2 + xs[i]^2)*m[i];
Iz[i]:=1/12*m[i]*(b[i]^2 + c[i]^2) + (xs[i]^2 + ys[i]^2)*m[i];
end if;
od;
```

The Equations for determining moments and products of inertia can be found in formularies, e.g. [Ref. 14]. No products of inertia were estimated. Reasons for that are given in [Ref. 8]. But there are products of inertia because the air-bearing design is not perfectly symmetrical and component locations cannot be adjusted very precisely. An estimation/measurement of these values should be done after all component properties are known.

Air-bearing Platform Moments of Inertia

The last sequence that can be seen below determines the air-bearing platform moments of inertia. They can be determined just by adding all component moments of inertia with command `sum`.

```
>  # i=1,2-xrods, i=3,4-yrods, i=5..10-jcells,
# i=11..14-zrods, i=15..18-angles, i=19-cpu_box,
# i=20-eps_box,i=21-magnetometer, i=22-wireless
# ethernet, i=23-disk, i=24-sexangle, i=25-bowl,
# i=26..29 ep_battaries
>  #print(Ix);
>  #print(Iy);
>  #print(Iz);
>  IX:=evalf(sum(Ix[j],'j'=1..N));
>  IY:=evalf(sum(Iy[j],'j'=1..N));
>  IZ:=evalf(sum(Iz[j],'j'=1..N));
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] *Space Systems Academic Group Homepage.*
http://www.sp.nps.navy.mil,
http://www.sp.nps.navy.mil/npsat1/education/educationpage.htm

[2] D.Sakoda, J.A.Horning Aug.2002.
Overview of the NPS Spacecraft Architecture and Technology Demonstration Satellite, NPSAT1.
http://www.sp.nps.navy.mil/npsat1/technical/technical_details.htm,
http://www.sp.nps.navy.mil/npsat1/technical/SSC02-I-4_Paper.pdf

[3] M.J. Sidi 1997.
Spacecraft Dynamics & Control.
Cambridge University Press, New York

[4] *International Geomagnetic Reference Field.*
http://nssdc.gsfc.nasa.gov/space/model/models/igrf.html

[5] B.S.Leonard Aug.2002.
NPSAT1 Magnetic Attitude Control System.
http://www.sp.nps.navy.mil/npsat1/technical/technical_details.htm,
http://www.sp.nps.navy.mil/npsat1/technical/SSC02-V-7.pdf

[6] B.S.Leonard.
class notes and notes concerning the simulation of the behavior of the air-bearing table
US Naval Postgraduate School

[7] M.Kohrt Apr.2002.
Embedding the NPSAT1 Attitude Control Subsystem (ACS) SIMULINK® Model.
Diplomarbeit, Space Systems Academic Group at US Naval Postgraduate School, Monterey, California

[8] A.Schmidt Jan.2003.
NPSAT1 Attitude Control Subsystem Hardware-in-the-Loop Simulation. Design of the air-bearing Platform.
Interdisziplinäre Studienarbeit, Space Systems Academic Group at US Naval Postgraduate School, Monterey, California

[9] Honeywell. *Smart Digital Magnetometer HMR2300.*
Datasheets.
http://www.ssec.honeywell.com/magnetic/datasheets/hmr2300.pdf

[10] Honeywell.
HMR2300 Smart Digital Magnetometer Demo Software / HMR2300 Smart Digital Magnetometer Demo Software Instructions.
http://www.ssec.honeywell.com:80/magnetic/interface/hmr2300_-demo.html

[11] *Meßtechnik. Manuskript zur Pflichtvorlesung für Studenten der Studienrichtungen Maschinenbau und Wirtschaftsingenieurwesen im sechsten Studientrimester.*
Universität der Bundeswehr Hamburg, Fachbereich Maschinenbau, Meß- und Informationstechnik, Mar.2001.

[12] R.Lunderstädt. 1990
Stochastik. Stochastische Systeme, Manuskript eines Teils der Vertiefungsvorlesung "Rechnerorientierte Verfahren der Regelungstechnik".
Universität der Bundeswehr Hamburg, Fachbereich Maschinenbau, Institut für Automatisierungstechnik

[13] W.Zeuge Aug.2001.
Skript zur Vorlesung Statistik.
Universität der Bundeswehr Hamburg, Fachbereich Maschinenbau, Mathematik

[14] H.Stöcker 1999.
Taschenbuch mathematischer Formeln und moderner Verfahren.
4.,korrigierte Auflage 1999, Verlag Harri Deutsch, Frankfurt am Main

[15] H.Nowotny Sep.2002.
Elektrodynamik und Relativitätstheorie. Anhang B: Einheitensysteme.
Technische Universität Wien, Institut für theoretische Physik
http://tph.tuwien.ac.at/ hnowotny/ws02ed.htm#script

[16] National Instruments.
NI Web Site. Support/Developer Zone.
http://www.ni.com/support/,
http://zone.ni.com/zone/jsp/zone.jsp

[17] National Instruments.
Writing a LabVIEW Instrument Driver. Knowledge Base.
http://www.ni.com/devzone/idnet/development.htm

[18] National Instruments.
LabVIEW Help. Instrument Drivers. Developing a Simple Driver.
Natinal Instruments, LabVIEW, November 2001 edition.

[19] National Instruments.
LabVIEW Help. / LabVIEW Context Help.
Natinal Instruments, LabVIEW, November 2001 edition.

[20] Mathworks.
xPC Documentations. xPC Target Getting Started Guide.
http://www.mathworks.com/access/helpdesk/help/toolbox/xpc/xpc.shtml, http://www.mathworks.com/access/helpdesk/help/pdf_doc/xpc/xpc_target_gs.pdf

[21] Mathworks.
xPC Documentations. xPC Target Users Guide.
http://www.mathworks.com/access/helpdesk/help/toolbox/xpc/xpc.shtml, http://www.mathworks.com/access/helpdesk/help/pdf_doc/xpc/xpc_target_ug.pdf

[22] Mathworks.
xPC Documentations. xPC Target I/O Reference Guide.
http://www.mathworks.com/access/helpdesk/help/toolbox/xpc/xpc.shtml, http://www.mathworks.com/access/helpdesk/help/pdf_doc/xpc/xpc_target_io_ref.pdf

[23] Mathworks.
Matlab Documentations. External Interfaces/API.
http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml, http://www.mathworks.com/access/helpdesk/help/pdf_-doc/matlab/apiext.pdf

[24] Mathworks.
Matlab Documentations. Matlab Functions - External Interfaces - Serial Port I/O Functions.
http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml

[25] Mathworks.
Simulink Documentations. Customizing SIMULINK - Writing S-Functions.
http://www.mathworks.com/access/helpdesk/help/toolbox/simulink/simulink.shtml, http://www.mathworks.com/access/helpdesk_r12p1/help/pdf_-doc/simulink/sfunctions.pdf

[26] Nelson Air Corp.
Naval Postgraduate School - 6" Spherical Air Bearing.
www.nelsonair.com
info@nelsonair.com

[27] MICROCOSM, Inc.
Magnetic Torquer MT30-2-CGS. Design Description and Operating Manual.
MMICROCOSM, Inc., Space Missions Enngineering
ElSegundo, CA90245

INITIAL DISTRIBUTION LIST

1.	Defense Technical Information Center 8725 John J. Kingman Rd., STE 0944 Ft. Belvoir, VA 22060-6218	2
2.	Dudley Knox Library, Code 013 Naval Postgraduate School Monterey, CA 93943-5100	2
3.	Research Office, Code 09 Naval Postgraduate School Monterey, CA 93943-5138	1